


Small Variance Asymptotics for Non-Parametric Online Robot Learning

The International Journal of Robotics Research
 XX(X):1–17
 ©The Author(s) 2016
 Reprints and permission:
 sagepub.co.uk/journalsPermissions.nav
 DOI: 10.1177/ToBeAssigned
 www.sagepub.com/


Ajay Kumar Tanwani^{1,2} and Sylvain Calinon¹

Abstract

Small variance asymptotics is emerging as a useful technique for inference in large scale Bayesian non-parametric mixture models. This paper analyses the online learning of robot manipulation tasks with Bayesian non-parametric mixture models under small variance asymptotics. The analysis yields a *scalable online sequence clustering* (SOSC) algorithm that is non-parametric in the number of clusters and the subspace dimension of each cluster. SOSC groups the new datapoint in its low dimensional subspace by online inference in a non-parametric *mixture of probabilistic principal component analyzers* (MPPCA) based on Dirichlet process, and captures the state transition and state duration information online in a *hidden semi-Markov model* (HSMM) based on hierarchical Dirichlet process. A task-parameterized formulation of our approach autonomously adapts the model to changing environmental situations during manipulation. We apply the algorithm in a teleoperation setting to recognize the intention of the operator and remotely adjust the movement of the robot using the learned model. The generative model is used to synthesize both time-independent and time-dependent behaviours by relying on the principles of shared and autonomous control. Experiments with the Baxter robot yield parsimonious clusters that adapt online with new demonstrations and assist the operator in performing remote manipulation tasks.

Keywords

Learning and Adaptive Systems, Bayesian Non-Parametrics, Online Learning, Subspace Clustering, Hidden Semi-Markov Model, Telerobotics, Teleoperation

1 Introduction

A long standing goal in artificial intelligence is to make robots interact with humans in everyday life tasks. Programming by demonstration provides a promising route to bridge this gap. When a set of T demonstrations of a manipulation task $\{\xi_t\}_{t=1}^T$ with $\xi_t \in \mathbb{R}^D$ is provided, it is often useful to encode these observations as a generative model with parameters θ (e.g., Gaussian mixture model or hidden Markov model), providing a probability density function $\mathcal{P}(\xi_t|\theta)$. With the influx of high-dimensional sensory data in robotics, an open challenge is to compactly encode such data online so that the robots are able to perform under varying environmental situations and across range of different tasks.

Bayesian non-parametric treatment of these clustering problems provides flexibility in model selection by maintaining an appropriate probability distribution over parameter values, $\mathcal{P}(\xi_t) = \int \mathcal{P}(\xi_t|\theta)\mathcal{P}(\theta)d\theta$. Although attractive for encapsulating *a priori* information about the task, the computational overhead of existing sampling-based and variational techniques for inference limit the widespread use of these models.

Recent analysis of Bayesian non-parametric mixture models under *small variance asymptotic* (SVA) limit has led to simple deterministic models that scale well with large size applications. For example, as the variances of the mixture model tend to zero in a GMM, the probabilistic

model converges to its deterministic counterpart, k -means, or to its non-parametric Dirichlet process (DP) version, DP-Means (Kulis and Jordan 2012). SVA analysis of other richer probabilistic models such as dependent DP mixture models (Campbell et al. 2013), hierarchical Dirichlet process (HDP) (Jiang et al. 2012), infinite latent feature models (Broderick et al. 2013), Markov jump processes (Huggins et al. 2015), infinite hidden Markov models (Roychowdhury et al. 2013), and infinite mixture of probabilistic principal component analysers (MPPCA) (Wang and Zhu 2015) leads to similar algorithms that scale well and yet retain the flexibility of non-parametric models.

1.1 Proposed Approach

In this paper, we investigate the online learning of robot manipulation tasks under SVA limit of Bayesian non-parametric models. We seek to incrementally update the parameters θ with each new observation ξ_{t+1} without having to retrain the model in a batch manner and store the demonstration data. To this end, we present an online inference algorithm for clustering sequential data, called *scalable online sequence clustering* (SOSC).

¹Idiap Research Institute, Switzerland.

²Ecole Polytechnique Federale de Lausanne (EPFL), Switzerland.

Corresponding author:

Ajay Kumar Tanwani

Email: ajay.tanwani@idiap.ch

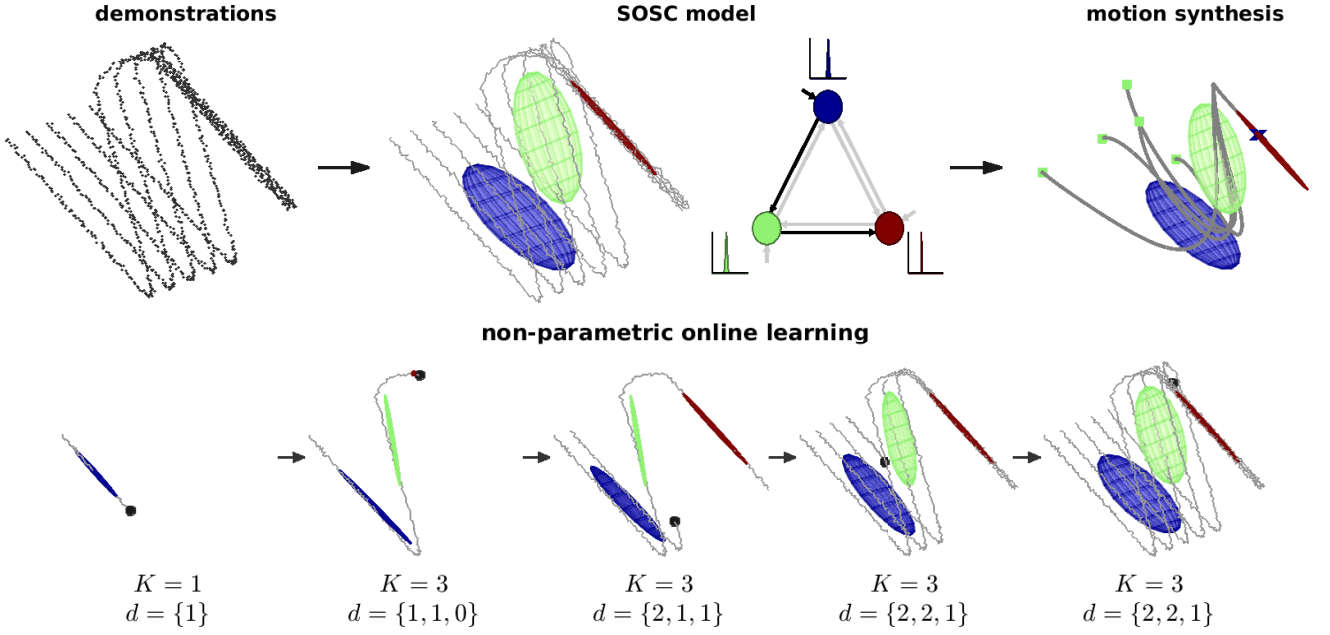


Figure 1. SOSC model illustration with Z-shaped streaming data composed of multiple trajectory samples. The model incrementally clusters the data in its intrinsic subspace. It tracks the transition among states and the state duration steps in a non-parametric manner. The generative model is used to recognize and synthesize motion in performing robot manipulation tasks (see Extension A-2).

SOSC incrementally groups the streaming data in its low-dimensional subspace by online inference in the Dirichlet process *mixture of probabilistic principal component analysers* (MPPCA) under small variance asymptotics, while being non-parametric in the number of clusters and the subspace dimension of each cluster. The model encapsulates the state transition and the state duration information in the data with online inference in a *hidden semi-Markov model* (HSMM) based on HDP. A task-parameterized formulation of the model is used to adapt the model parameters to varying environmental situations in a probabilistic manner (Calinon 2016; Tanwani and Calinon 2016). The proposed approach uses the learning from demonstrations paradigm to teach manipulation tasks to robots in an online and intuitive manner. We show its application in a teleoperation scenario where the SOSC model is built online from the demonstrations provided by the teleoperator to perform remote robot manipulation tasks (see Fig. 1 for an overview of our approach).

1.2 Contributions

The purpose of this paper is to present an online unsupervised learning framework that is fast and scalable for the encoding of a large range of robot manipulation tasks in a non-parametric manner. The contributions of the paper are:

- Online inference algorithms for DP-MPPCA and HDP-HSMM under small variance asymptotics,
- Non-parametric SOSC algorithm for online learning and motion synthesis of high-dimensional robot manipulation tasks,
- Task-parameterized formulation of the SOSC model to systematically adapt the model parameters to changing situations such as position/orientation/size of the objects,

- Extension of learning from demonstration to the context of semi-autonomous teleoperation.

Organization of the paper: We give a brief overview of unsupervised learning approaches for robot learning in Sec. 2. Sec. 3 formalizes our learning problem followed by online inference algorithms of DP-MPPCA and HDP-HSMM in Sec. 4 and Sec. 5 respectively. In Sec. 6, we present the SOSC algorithm and then evaluate its performance on synthetic data and semi-autonomous teleoperation with the Baxter robot in Sec. 7. Finally, we conclude the paper with an outlook to our future work.

2 Background and Related Work

SOSC builds a generative model of the demonstrations online by clustering the streaming data in its low-dimensional subspace and capturing the state transition and state duration information in a non-parametric manner. Incremental online learning poses a unique challenge to the existing robot learning methods with high-dimensional data, model selection, real-time adaptation and adequate accuracy/generalization after observing a fewer number of training samples. An overview of the robot learning from demonstration methods can be found in (Schaal et al. 2003; Argall et al. 2009; Stulp and Sigaud 2015; Billard et al. 2016).

Gaussian mixture models (GMMs) are widely used to encode local trends in the demonstrations for unsupervised learning problems. The probability density function \mathcal{P} of a GMM with K mixture components is represented as

$$\mathcal{P}(\xi_t|\theta_r) = \sum_{i=1}^K \pi_i \mathcal{N}(\xi_t|\mu_i, \Sigma_i), \quad (1)$$

where $\mathcal{N}(\mu_i, \Sigma_i)$ is the multivariate Gaussian distribution with parameters θ_r containing prior $\pi_i \in \mathbb{R}$, mean $\mu_i \in \mathbb{R}^D$, and covariance matrix $\Sigma_i \in \mathbb{R}^{D \times D}$. **Hidden Markov models (HMMs)** encapsulate the spatio-temporal information by augmenting a GMM with latent states that sequentially evolve over time in the demonstrations. The parameter set now additionally contains the transition matrix and the initial state distribution. HMMs are widely used for time series/sequence analysis in speech recognition, machine translation, DNA sequencing, robotics and many other fields (Rabiner 1989). HMMs have been typically used for recognition and generation of movement skills in robotics (Asfour et al. 2008; Calinon et al. 2010; Lee et al. 2010; Vakanski et al. 2012). A number of variants of HMMs have been proposed to address some of its shortcomings, including: 1) how to bias learning towards models with longer self-dwelling states, 2) how to robustly estimate the parameters with high-dimensional noisy data, 3) how to adapt the model with newly observed data, and 4) how to estimate the number of states that the model should possess.

Variants based on **Hidden semi-Markov models (HSMMs)** replace the self-transition probabilities of staying in a state with an explicit model of state duration (Yu 2010). This helps to adequately bias the generated motion with longer state dwell times for learning robot manipulation tasks (Tanwani and Calinon 2016).

Subspace clustering methods perform segmentation and dimensionality reduction simultaneously to encode human demonstrations with piecewise planar segments (Schaal et al. 2007). Statistical subspace clustering methods impose a parsimonious structure on the covariance matrix to reduce the number of parameters that can be robustly estimated (Bouveyron and Brunet 2014). For example, a *mixture of factor analyzers* (MFA) performs subspace clustering by assuming the structure of the covariance to be of the form (McLachlan et al. 2003)

$$\Sigma_i = \Lambda_i^d \Lambda_i^{d^T} + \Psi_i, \quad (2)$$

where $\Lambda_i^d \in \mathbb{R}^{D \times d}$ is the *factor loadings matrix* with $d < D$ for parsimonious representation of the data, and $\Psi_i \in \mathbb{R}^{D \times D}$ is the diagonal noise matrix. Other extensions such as sharing the parameters of the covariance in a semi-tied manner aligns the mixture components for robust encoding of the demonstrations (Tanwani and Calinon 2016).

Online/Incremental learning methods update the model parameters with streaming data, without the need to re-train the model in a batch manner (Neal and Hinton 1999; Song and Wang 2005). Non-parametric regression methods have been commonly used in this context such as locally weighted projection regression (Vijayakumar et al. 2005), sparse online Gaussian process regression (Gijbarts and Metta 2013) and their fusion with local Gaussian process regression (Nguyen-Tuong et al. 2009). Kulic et al. (2008) used HMMs to incrementally group whole-body motions based on their relative distance in HMM space. Lee and Ott (2010) presented an iterative motion primitive refinement approach with HMMs. Kronander et al. (2015) locally reshaped an existing dynamical system with new demonstrations in an incremental manner while preserving its stability. Hoyos et al. (2016) experimented with different strategies to incrementally add demonstrations

to a task-parametrized GMM. Bruno et al. (2016) learned autonomous behaviours for a flexible surgical robot by online clustering with DP-means.

Bayesian non-parametric treatment of HMMs/HSMMs automates the number of states selection procedure by Bayesian inference in a model with infinite number of states (Beal et al. 2002; Johnson and Willsky 2013). Niekum et al. (2012) used the Beta Process Autoregressive HMM for learning from unstructured demonstrations. Krishnan et al. (2015) defined a hierarchical non-parametric Bayesian model to identify the transition structure between states with a linear dynamical system. Inferring the maximum *a posteriori* distribution of the parameters in non-parametric models, however, is often difficult. Markov Chain Monte Carlo (MCMC) sampling or variational methods are required which are difficult to implement and often do not scale with the size of the data. Small variance asymptotic analysis of these methods provide a trade-off by yielding simple and scalable hard clustering non-parametric algorithms (Kulis and Jordan 2012). Other prominent applications of SVA include feature learning (Broderick et al. 2013), dimensionality reduction (Wang and Zhu 2015) and time-series analysis (Roychowdhury et al. 2013).

This paper builds upon these advancements in small variance asymptotic analysis of Bayesian non-parametric mixture models. We present a non-parametric online unsupervised framework for robot learning from demonstrations, which scales well with sequential high-dimensional data. We formulate online inference algorithms of HDP-HSMM and DP-MPPCA under small variance asymptotics. We then learn a task-parameterized generative model online for encoding and motion synthesis of robot manipulation tasks.

3 Problem Setup

Let us consider the streaming observation sequence $\{\xi_1, \dots, \xi_t\}$ with $\xi_t \in \mathbb{R}^D$ obtained at current time step t while demonstrating a manipulation task. The corresponding hidden state sequence $\{z_1, \dots, z_t\}$ with $z_t \in \{1, \dots, K\}$ belongs to the discrete set of K cluster indices at time t . Let $\mathbf{a} \in \mathbb{R}^{K \times K}$ with $a_{i,j} \triangleq P(z_t = j | z_{t-1} = i)$ denote the transition probability of moving from state i at time $t-1$ to state j at time t . The parameters $\{\mu_j^S, \Sigma_j^S\}$ represent the mean and the standard deviation of staying s consecutive time steps in state j estimated by a Gaussian $\mathcal{N}(s | \mu_j^S, \Sigma_j^S)$. The hidden state follows a multinomial distribution with $z_t \sim \text{Mult}(\pi_{z_{t-1}})$ where $\pi_{z_{t-1}} \in \mathbb{R}^K$ is the next state transition distribution over state z_{t-1} , and the observation ξ_t is drawn from the output distribution of state j , described by a multivariate Gaussian with parameters $\{\mu_j, \Sigma_j\}$. The K Gaussian components constitute a GMM augmented with the state transition and the state duration model to capture the sequential pattern in the demonstrations (see Appx. B for the notations used in the paper).

Note that if the covariance matrices Σ_j of all the mixture components in a GMM are set equal to the isotropic matrix $\sigma^2 \mathbf{I}$, the auxiliary function of GMM, $Q(\theta_r, \theta_r^{\text{old}}) = \mathbb{E} \{ \log \mathcal{P}(\xi_t, z_t | \theta_r) | \xi_t, \theta_r^{\text{old}} \}$, takes

the form (Dempster et al. 1977)

$$\sum_{i=1}^K \mathcal{P}(i|\xi_t, \theta_r^{\text{old}}) \left(\log \pi_i - \frac{D}{2} \log 2\pi\sigma^2 - \frac{\|\xi_t - \mu_i\|_2^2}{2\sigma^2} \right). \quad (3)$$

Applying the small variance asymptotic limit to the auxiliary function with $\lim_{\sigma^2 \rightarrow 0} \mathcal{Q}(\theta_r, \theta_r^{\text{old}})$, the last term $\frac{\|\xi_t - \mu_i\|_2^2}{2\sigma^2}$ dominates the objective function and the maximum likelihood estimate reduces to the k -means problem,* i.e.,

$$\max_{\theta_r} \mathcal{Q}(\theta_r, \theta_r^{\text{old}}) = \arg \min_{z_t, \mu} \|\xi_t - \mu_{z_t}\|_2^2. \quad (4)$$

By restricting the covariance matrix to an isotropic/spherical noise, the number of parameters grows up to a constant with the dimension of datapoint D . Although attractive for scalability and parsimonious structure, such decoupling cannot encode the important motor control principles of coordination, synergies and action-perception couplings.

Consequently, we further assume that the j^{th} output Gaussian groups the observation ξ_t in its intrinsic low-dimensional affine subspace of dimension d_j with projection matrix $\Lambda_j^{d_j} \in \mathbb{R}^{D \times d_j}$, such that $d_j < D$ and $\Sigma_j = \Lambda_j^{d_j} \Lambda_j^{d_j^\top} + \sigma_j^2 \mathbf{I}$. Under this assumption, we apply the small variance asymptotic limit on the remaining $(D - d_j)$ dimensions to encode the most important coordination patterns while being parsimonious in the number of parameters. The overall parameter set is represented by $\theta_h = \{\mu_i, \Sigma_i, \{a_{i,m}\}_{m=1}^K, \mu_i^S, \Sigma_i^S\}_{i=1}^K$.† We are interested in updating the parameter set θ_h online upon observation of a new datapoint ξ_{t+1} , such that the datapoint can be discarded afterwards. We apply the Bayesian non-parametric treatment to this model and formulate online inference algorithms for DP-MPPCA and HDP-HSMM under small variance asymptotics. This provides a non-parametric online approach to robot learning from demonstrations.

4 Online DP-MPPCA

In this section, we restrict our problem formulation to encode the output state distributions of the model with a *mixture of probabilistic principal component analyzers* (MPPCA) using the parameter subset $\theta_s = \{\mu_i, \Lambda_i^d, d_i\}_{i=1}^K$. We consider its non-parametric extension with the Dirichlet process under small variance asymptotics and present an algorithm for online inference.

4.1 Mixture of Probabilistic Principal Component Analyzers (MPPCA)

The basic idea of MPPCA is to reduce the dimensions of the data while keeping the observed covariance structure. The generative model of MPPCA approximates the datapoint ξ_t as a convex combination of K subspace clusters (Tipping and Bishop 1999),

$$\mathcal{P}(\xi_t|\theta_s) = \sum_{i=1}^K \mathcal{P}(z_t = i) \mathcal{N}(\xi_t|\mu_i, \Lambda_i^d \Lambda_i^{d^\top} + \sigma_i^2 \mathbf{I}), \quad (5)$$

where $\mathcal{P}(z_t = i)$ is the cluster prior, $\Lambda_i^d \in \mathbb{R}^{D \times d}$ is the projection matrix with $d < D$ and $d = d_i$, $\sigma_i^2 \mathbf{I}$ is the

isotropic noise coefficient for the i -th cluster, and the covariance structure is of the form $\Sigma_i = \Lambda_i^d \Lambda_i^{d^\top} + \sigma_i^2 \mathbf{I}$.‡ The model assumes that ξ_t , conditioned on $z_t = i$, is generated by an affine transformation of d -dimensional latent variable $u_t \in \mathbb{R}^d$ with noise term $\epsilon \in \mathbb{R}^D$ such that

$$\xi_t = \Lambda_i^d u_t + \mu_i + \epsilon, \quad u_t \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_d), \quad \epsilon \sim \mathcal{N}(\mathbf{0}, \sigma_i^2 \mathbf{I}). \quad (6)$$

The model parameters of MPPCA are usually learned using an Expectation-Maximization (EM) procedure (Tipping and Bishop 1999). But in this case, both the number of clusters K and the subspace dimension of each cluster d needs to be specified a priori which is not always trivial in several domains.

4.2 Dirichlet Process MPPCA (DP-MPPCA)

Bayesian non-parametric extension of MPPCA alleviates the problem of model selection by defining prior distributions over the number of clusters K and the subspace dimension of each cluster d_i (Zhang et al. 2004; Chen et al. 2010; Wang and Zhu 2015).

Following (Wang and Zhu 2015), a *Chinese Restaurant Process* (CRP) prior is placed over the cluster assignment z_t , and a hierarchical prior is imposed on the projection matrix $\Lambda_i^{d_i}$ with an exponential prior on the subspace rank d_i . Performing small variance asymptotics on the resulting partially collapsed Gibbs sampler leads to an efficient deterministic algorithm for subspace clustering with an infinite MPPCA. The algorithm iteratively converges by minimizing the loss function

$$\mathcal{L}(z, d, \mu, U) = \lambda K + \lambda_1 \sum_{i=1}^K d_i + \sum_{t=1}^T \text{dist}(\xi_t, \mu_{z_t}, U_{z_t}^d)^2, \quad (7)$$

where $\text{dist}(\xi_t, \mu_{z_t}, U_{z_t}^d)^2$ represents the distance of the datapoint ξ_t to the subspace of cluster z_t defined by mean μ_{z_t} and unit eigenvectors of the covariance matrix $U_{z_t}^d$ (see Eq. (8)), and λ, λ_1 represent the penalty terms for the number of clusters and the subspace dimension of each cluster respectively. The algorithm optimizes the number of clusters and the subspace dimension of each cluster while minimizing the distance of the datapoints to the respective subspaces of each cluster.

4.3 Online Inference in DP-MPPCA

In the online setting, we seek to incrementally update the parameters $\theta_{t,s}$ (θ_s at time t) with the new observation ξ_{t+1} without having to retrain the model in a batch manner and store the demonstration data. The parameters are updated in an online manner in two steps: the cluster assignment step followed by the parameter updates step.

*SVA analysis of the Bayesian non-parametric GMM leads to the DP-means algorithm (Kulis and Jordan 2012). Similarly, SVA analysis of the HMM yields the segmental k -means problem (Roychowdhury et al. 2013).

†With a slight abuse of notation, we represent the parameters with an added subscript t for online learning. For example, $\theta_{t,h}$ denotes the parameters of θ_h at time t .

‡Note that MPPCA is closely related to MFA, and uses isotropic noise matrix instead of the diagonal noise matrix used in MFA (see Eq. (2)).

4.3.1 Cluster Assignment z_{t+1} : The basic rationale in the cluster assignment step is to substitute the probability of the datapoint ξ_{t+1} to belong to the i -th mixture component with the distance of the datapoint to the existing subspaces of the clusters. If the distance to the farthest subspace is greater than a certain threshold λ , a new cluster is initialized with that datapoint; otherwise the datapoint is assigned to the cluster with the least distance to its subspace. We compute the distance of the datapoint to the subspace of the i -th cluster as

$$\text{dist}(\xi_{t+1}, \mu_{t,i}, U_{t,i}^{d_{t,i}}) = \left\| (\xi_{t+1} - \mu_{t,i}) - \rho_i U_{t,i}^{d_{t,i}} U_{t,i}^{d_{t,i}\top} (\xi_{t+1} - \mu_{t,i}) \right\|_2, \quad (8)$$

where

$$\rho_i = \exp \left(-\frac{\|\xi_{t+1} - \mu_{t,i}\|_2^2}{b_m} \right)$$

weighs the projected mean-centered datapoint according to the distance of the datapoint from the cluster center ($0 < \rho_i \leq 1$). Its effect is controlled by the bandwidth parameter b_m . Large values of b_m mean that clusters far away have a greater influence, while small bandwidths favour nearby clusters only. Note that our subspace distance formulation is different from (Wang and Zhu 2015) as we weigh the subspace of the nearby clusters more than the distant clusters to limit the size of the subspace. The cluster assignment is deterministically updated using

$$z_{t+1} = \arg \min_{i=1:K+1} \begin{cases} \text{dist}(\xi_{t+1}, \mu_{t,i}, U_{t,i}^{d_{t,i}})^2, & \text{if } i \leq K \\ \lambda, & \text{otherwise.} \end{cases} \quad (9)$$

4.3.2 Parameter Updates $\theta_{t+1,s}$: Given the cluster assignment $z_{t+1} = i$ at time $t+1$, prior $\pi_{t+1,i}$ and mean $\mu_{t+1,i}$ of the i -th cluster are updated as

$$\begin{aligned} \pi_{t+1,i} &= \frac{1}{t+1} (t\pi_{t,i} + 1), \\ \mu_{t+1,i} &= \frac{1}{w_{t,i} + 1} (w_{t,i}\mu_{t,i} + \xi_{t+1}), \end{aligned} \quad (10)$$

where $w_{t,i}$ is the weight assigned to the i -th cluster parameter set at time t to control the effect of the parameter update with the new datapoint at time $t+1$ relative to the updates seen till time t . Depending upon the nature of the streaming data, $w_{t+1,i}$ can be updated as follows:

- For stationary online learning problems where the data is sampled from some fixed distribution, we update the weight $w_{t+1,i}$ linearly with the number of instances belonging to that cluster, namely

$$w_{t+1,i} = w_{t,i} + 1, \quad w_{0,i} = 1. \quad (11)$$

- For non-stationary online learning problems where the distribution of streaming data varies over time, we update the weight vector based on the *eligibility trace* that takes into account the temporary occurrence of visiting a particular cluster.[§] The trace indicates how much a cluster is eligible for undergoing changes with the new parameter update. The trace is updated such

that the weights of all the clusters are decreased by the discount factor $\zeta \in (0, 1)$ and the weight of the visited cluster is incremented, i.e., the more often a state is visited, the higher is the eligibility weight of all the previous updates relative to the new parameter update, namely

$$w_{t+1,i} = \begin{cases} \zeta w_{t,i} + 1, & \text{if } i = z_{t+1} \\ \zeta w_{t,i}, & \text{if } i \neq z_{t+1}. \end{cases} \quad (12)$$

- For non-stationary problems where learning is continuous and may not depend upon the number of datapoints, the weight vector is kept constant $w_{t+1,i} = w_{t,i} = w^*$ at all time steps as a step-size parameter.

The covariance matrix could then be updated online as

$$\begin{aligned} \Sigma_{t+1,i} &= \frac{w_{t,i}}{w_{t,i} + 1} \Sigma_{t,i} + \\ &\quad \frac{w_{t,i}}{(w_{t,i} + 1)^2} (\xi_{t+1} - \mu_{t+1,i})(\xi_{t+1} - \mu_{t+1,i})^\top. \end{aligned} \quad (13)$$

However, updating the covariance matrix online in D -dimensional space can be prohibitively expensive for even moderate size problems. To update the covariance matrix in its intrinsic lower dimension, similarly to (Bellas et al. 2013), we compute $g_{t+1,i} \in \mathbb{R}^{d_i}$ as the projection of datapoint ξ_{t+1} onto the existing set of basis vectors of $U_{t,i}^{d_{t,i}}$. Note that the cardinality of basis vectors is different for each covariance matrix. If the datapoint belongs to the subspace of $U_{t,i}^{d_{t,i}}$, the retro-projection of the datapoint in its original space as given by the residual vector $p_{t+1,i} \in \mathbb{R}^D$ would be a zero vector; otherwise the residual vector belongs to the null space of $U_{t,i}^{d_{t,i}}$, and its unit vector $\tilde{p}_{t+1,i}$ needs to be added to the existing set of basis vectors, i.e.,

$$\begin{aligned} g_{t+1,i} &= U_{t,i}^{d_{t,i}\top} (\xi_{t+1} - \mu_{t,i}), \\ p_{t+1,i} &= (\xi_{t+1} - \mu_{t,i}) - U_{t,i}^{d_{t,i}} g_{t+1,i}, \\ \tilde{p}_{t+1,i} &= \begin{cases} \frac{p_{t+1,i}}{\|p_{t+1,i}\|_2}, & \text{if } \|p_{t+1,i}\|_2 > 0 \\ 0_D, & \text{otherwise.} \end{cases} \end{aligned}$$

The augmented set of basis vectors are represented as

$$U_{t+1,i}^{d_{t,i}} = [U_{t,i}^{d_{t,i}}, \tilde{p}_{t+1,i}] R_{t+1,i}, \quad (14)$$

where, $R_{t+1,i} \in \mathbb{R}^{(d_{t,i}+1) \times (d_{t,i}+1)}$ is the rotation matrix to incrementally update the augmented basis vectors. $R_{t+1,i}$ is obtained by simplifying the eigendecomposition problem

$$\Sigma_{t+1,i} = U_{t+1,i}^{d_{t,i}} \Sigma_{t+1,i}^{(\text{diag})} U_{t+1,i}^{d_{t,i}\top}. \quad (15)$$

Substituting the value of $\Sigma_{t+1,i}$ and $U_{t+1,i}^{d_{t,i}}$ from Eq. (13) and (14) yields the reduced eigendecomposition problem of size $(d_{t,i} + 1) \times (d_{t,i} + 1)$ with

$$\begin{aligned} &\frac{w_{t,i}}{w_{t,i} + 1} \begin{bmatrix} \Sigma_{t,i}^{(\text{diag})} & \mathbf{0}_{d_{t,i}} \\ \mathbf{0}_{d_{t,i}}^\top & 0 \end{bmatrix} + \frac{w_{t,i}}{(w_{t,i} + 1)^2} \\ &\begin{bmatrix} g_{t+1,i} g_{t+1,i}^\top & \nu_i g_{t+1,i} \\ \nu_i g_{t+1,i}^\top & \nu_i^2 \end{bmatrix} = R_{t+1,i} \Sigma_{t+1,i}^{(\text{diag})} R_{t+1,i}^\top, \end{aligned} \quad (16)$$

[§]Eligibility traces are commonly used in reinforcement learning to evaluate the state for undergoing learning changes in temporal-difference learning (Sutton and Barto 1998).

where $\nu_i = \tilde{\mathbf{p}}_{t+1,i}^\top (\boldsymbol{\xi}_{t+1} - \boldsymbol{\mu}_{t+1,i})$. Solving for $\mathbf{R}_{t+1,i}$ and substituting it in Eq. (14) gives the required updates of the basis vectors in a computationally and memory efficient manner. The subspace dimension of the i -th mixture component is updated by keeping an estimate of the average distance vector $\bar{\mathbf{e}}_{t,i} \in \mathbb{R}^D$ whose k -th element represents the mean distance of the datapoints to the $(k-1)$ subspace basis vectors of $\mathbf{U}_{t,i}^k$ for the i -th cluster. Let us denote $\boldsymbol{\delta}_i$ as the vector measuring the distance of the datapoint $\boldsymbol{\xi}_{t+1}$ to each of the subspaces of $\mathbf{U}_{t,i}^k$ for the i -th cluster where $k = \{0 \dots (d_{t,i} + 1)\}$, i.e.,

$$\boldsymbol{\delta}_i = \begin{bmatrix} \text{dist}(\boldsymbol{\xi}_{t+1}, \boldsymbol{\mu}_{t+1,i}, \mathbf{U}_{t+1,i}^0)^2 \\ \vdots \\ \text{dist}(\boldsymbol{\xi}_{t+1}, \boldsymbol{\mu}_{t+1,i}, \mathbf{U}_{t+1,i}^{d_{t,i}+1})^2 \end{bmatrix}, \quad (17)$$

where $\text{dist}(\boldsymbol{\xi}_{t+1}, \boldsymbol{\mu}_{t+1,i}, \mathbf{U}_{t+1,i}^0)^2$ is the distance to the cluster subspace with 0 dimension (the cluster center point), $\text{dist}(\boldsymbol{\xi}_{t+1}, \boldsymbol{\mu}_{t+1,i}, \mathbf{U}_{t+1,i}^1)^2$ is the distance to the cluster subspace with 1 dimension (the line), and so on. The average distance vector $\bar{\mathbf{e}}_{t+1,i}$ and the subspace dimension $d_{t+1,i}$ are updated as

$$\bar{\mathbf{e}}_{t+1,i} = \frac{1}{w_{t,i} + 1} (w_{t,i} \bar{\mathbf{e}}_{t,i} + \boldsymbol{\delta}_i), \quad (18)$$

$$d_{t+1,i} = \arg \min_{d=0:D-1} \left\{ \lambda_1 d + \bar{\mathbf{e}}_{t+1,i} \right\}. \quad (19)$$

Given the updated set of basis vectors, the projection matrix and the covariance matrix are updated as

$$\boldsymbol{\Lambda}_{t+1,i}^{d_{t+1,i}} = \mathbf{U}_{t+1,i}^{d_{t+1,i}} \sqrt{\boldsymbol{\Sigma}_{t+1,i}^{(\text{diag})}}, \quad (20)$$

$$\boldsymbol{\Sigma}_{t+1,i} = \boldsymbol{\Lambda}_{t+1,i}^{d_{t+1,i}} \boldsymbol{\Lambda}_{t+1,i}^{d_{t+1,i}\top} + \sigma^2 \mathbf{I}. \quad (21)$$

The loss function optimized at time step $t+1$ is estimated as

$$\begin{aligned} \mathcal{L}(z_{t+1}, d_{t+1,z_{t+1}}, \boldsymbol{\mu}_{t+1,z_{t+1}}, \mathbf{U}_{t+1,z_{t+1}}^{d_{t+1,z_{t+1}}}) &= \lambda K + \\ \lambda_1 d_{t+1,z_{t+1}} + \text{dist}(\boldsymbol{\xi}_{t+1}, \boldsymbol{\mu}_{t+1,z_{t+1}}, \mathbf{U}_{t+1,z_{t+1}}^{d_{t+1,z_{t+1}}})^2 \\ &\leq \mathcal{L}(z_{t+1}, d_{t,z_{t+1}}, \boldsymbol{\mu}_{t,z_{t+1}}, \mathbf{U}_{t,z_{t+1}}^{d_{t,z_{t+1}}}). \end{aligned}$$

The loss function provides an intuitive trade-off between the fitness term $\text{dist}(\boldsymbol{\xi}_{t+1}, \boldsymbol{\mu}_{t+1,z_{t+1}}, \mathbf{U}_{t+1,z_{t+1}}^{d_{t+1,z_{t+1}}})^2$ and the model selection parameters K and d_k . Increasing the number of clusters or the subspace dimension of the assigned cluster decreases the distance of the datapoint to the assigned subspace at the cost of penalty terms λ and λ_1 . Parameters of the assigned cluster are updated in a greedy manner such that the loss function is guaranteed to decrease at the current time step.

5 Online HDP-HSMM

In this section, we first briefly describe HSMM and its Bayesian non-parametric extension, and then present our incremental formulation to estimate the parameters of an infinite HSMM, $\theta_h = \left\{ \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i, \{a_{i,m}\}_{m=1}^K, \mu_i^S, \Sigma_i^S \right\}_{i=1}^K$, where the output distribution of i -th state is represented

by a parsimonious multivariate Gaussian $\mathcal{N}(\boldsymbol{\mu}_i, \boldsymbol{\Lambda}_i^{d_i} \boldsymbol{\Lambda}_i^{d_i\top} + \sigma^2 \mathbf{I})$. The additional parameters from the last section here correspond to the transition probabilities and the explicit state duration model for each state.

5.1 Hidden Semi-Markov Model (HSMM)

A hidden Markov model describes a latent Markov process that transitions between a finite number of states at discrete times while emitting an observation in each state. Spatio-temporal encoding with HMMs can handle movements with variable durations, recurring patterns, options in the movement, or partial/unaligned demonstrations. Learning in HMMs usually requires experimenting with the structure of transitions and the number of latent states. For example, left-to-right HMMs preclude all the states previously visited by setting constraints to the corresponding transition probabilities to be zero.

A hidden semi-Markov model (HSMM) relaxes the Markovian structure of state transitions by relying not only upon the current state but also on the duration/elapsed time in the current state. An explicit duration HSMM sets the self-transition probabilities to zero and explicitly models the state duration with a parametric distribution (Yu 2010) (we use a Gaussian distribution here to model the state duration). Though HMMs also implicitly assume that the duration of staying in a state follows a geometric distribution, this assumption is often limiting, especially for the modeling of sequences with long state dwell-times (Rabiner 1989).

5.2 Hierarchical Dirichlet Process Hidden Semi-Markov Model (HDP-HSMM)

Specifying the number of latent states in a HMM/HSMM is often difficult. Model selection methods such as cross-validation or Bayesian Information Criterion (BIC) are typically used to determine the number of states. Bayesian non-parametric approaches comprising of HDPs provide a principled model selection procedure by Bayesian inference in a HMM/HSMM with infinite number of states. Interested readers can find details of DPs and HDPs for specifying an infinite set of conditional transition distribution priors in Teh et al. (2006).

HDP-HMM (Beal et al. 2002; Van Gael et al. 2008) is an infinite state Bayesian non-parametric generalization of the HMM with HDP prior on the transition distribution. In this model, the state transition distribution for each state follows a Dirichlet process $\mathbf{G}_i \sim \text{DP}(\alpha, \mathbf{G}_0)$ with concentration parameter α and shared base distribution \mathbf{G}_0 such that \mathbf{G}_0 is the global Dirichlet process $\mathbf{G}_0 \sim \text{DP}(\gamma, \mathbf{H})$ with concentration parameter γ and base distribution \mathbf{H} . The top level DP enables sharing of the existing states with a new state created under a bottom level DP for each state and encourages visiting of the same consistent set of states in the sequence. Let $\boldsymbol{\beta}$ denote the weights of \mathbf{G}_0 in its

stick-breaking construction (Sethuraman 1994), then the non-parametric approach takes the form

$$\begin{aligned}\beta|\gamma &\sim \text{GEM}(\gamma), \\ \pi_i|\alpha, \beta &\sim \text{DP}(\alpha, \beta), \\ \{\mu_i, \Lambda_i^{d_i}, d_i\} &\sim H, \\ z_t &\sim \text{Mult}(\pi_{z_{t-1}}), \\ \xi_t|z_t &\sim \mathcal{N}(\mu_{z_t}, \Lambda_{z_t}^{d_{z_t}} \Lambda_{z_t}^{d_{z_t}^\top} + \sigma^2 I),\end{aligned}$$

where GEM represents the Griffiths, Engen and McCloskey distribution (Pitman 2002), and we have used the parsimonious representation of a Gaussian for the output distribution of a state without loss of generality.

Johnson and Willsky (2013) presented an extension of HDP-HMM to HDP-HSMM by explicitly drawing the state duration distribution parameters and precluding the self-transitions. Other extensions such as sticky HDP-HMM (Fox et al. 2008) add a self-transition bias parameter to the DP of each state to prolong the state-dwell times. We take a simpler approach to explicitly encode the state duration by setting the self-transition probabilities to zero and estimating the parameters $\{\mu_i^S, \Sigma_i^S\}$ empirically from the hidden state sequence $\{z_1, \dots, z_T\}$.

Note that learning the model in this Bayesian non-parametric setting involves computing the posterior distribution over the latent state, the output state distribution and the transition distribution parameters. The problem is more challenging than the maximum likelihood parameter estimation of HMMs and requires MCMC sampling or variational inference techniques to compute the posterior distribution. Performing small-variance asymptotics of the joint likelihood of HDP-HMM, on the other hand, yields the maximum *a posteriori* estimates of the parameters that iteratively minimize the loss function[¶]

$$\begin{aligned}\mathcal{L}(z, d, \mu, U, a) &= \sum_{t=1}^T \text{dist}(\xi_t, \mu_{z_t}, U_{z_t}^{d_{z_t}})^2 + \lambda(K-1) \\ &+ \lambda_1 \sum_{i=1}^K d_i - \lambda_2 \sum_{t=1}^{T-1} \log(a_{z_t, z_{t+1}}) + \lambda_3 \sum_{i=1}^K (\tau_i - 1),\end{aligned}$$

where $\lambda_2, \lambda_3 > 0$ are the additional penalty terms responsible for prolonging the state duration estimates compared to the loss function in Eq. (7). The λ_2 term favours the transitions to states with higher transition probability (states which have been visited more often before), λ_3 penalizes for transition to unvisited states with τ_i denoting the number of distinct transitions out of state i , and λ, λ_1 are the penalty terms for increasing the number of states and the subspace dimension of each output state distribution.

5.3 Online Inference in HDP-HSMM

For the online setting, we denote the parameter set θ_h at time t as $\theta_{t,h}$. Given the observation ξ_{t+1} , we now present the cluster assignment and the parameter update steps for the online incremental version of HDP-HSMM.

5.3.1 Cluster Assignment z_{t+1} : The datapoint ξ_{t+1} is assigned to cluster z_{t+1} based on the rule

$$z_{t+1} = \arg \min_{i=1:K+1} \begin{cases} q_{1,i}, & \text{if } \{a_{t,z_t,i} > 0, i \leq K\} \\ q_{2,i}, & \text{if } \{a_{t,z_t,i} = 0, i \leq K\} \\ q_{3,i}, & \text{otherwise,} \end{cases} \quad (22)$$

$$q_{1,i} = \text{dist}(\xi_{t+1}, \mu_{t,i}, U_{t,i}^{d_{t,i}})^2 - \lambda_2 \log a_{t,z_t,i}, \quad (22)$$

$$q_{2,i} = \text{dist}(\xi_{t+1}, \mu_{t,i}, U_{t,i}^{d_{t,i}})^2 - \lambda_2 \log \frac{1}{\sum_{k=1}^K c_{t,z_t,k} + 1} + \lambda_3, \quad (23)$$

$$q_{3,i} = \lambda - \lambda_2 \log \frac{1}{\sum_{k=1}^K c_{t,z_t,k} + 1} + \lambda_3, \quad (24)$$

where $c_{t,i,j}$ is an auxiliary transition variable that counts the number of visits from state i to state j till time t . The assignment procedure evaluates the cost on two main criteria: 1) distance of the datapoint to the existing cluster subspaces given by $\text{dist}(\xi_{t+1}, \mu_{t,i}, U_{t,i}^{d_{t,i}})$, and 2) transition probability of moving from the current state to the other state $a_{t,z_t,i}$. The procedure favours the next state to be one whose distance from the subspace of a cluster is low and whose transition probability is high, as seen in Eq. (22). If the probability of transitioning to a given state is zero, an additional penalty of λ_3 is added along with a pseudo transition count to that state $\frac{1}{\sum_{k=1}^K c_{t,z_t,k} + 1}$. Finally, if the cost of transitioning to a new state at subspace distance λ in Eq. (24) is lower than the cost evaluated in Eq. (22) and Eq. (23), a new cluster is created with the datapoint and default parameters.

5.3.2 Parameter Updates $\theta_{t+1,h}$: Given the cluster assignment $z_{t+1} = i$, we first estimate the parameters $\mu_{t+1,i}, U_{t+1,i}^{d_{t+1,i}}, d_{t+1,i}$, and $\Sigma_{t+1,i}$ following the update rules in Eqs (10), (14), (19) and (21), respectively. We update the transition probabilities via the auxiliary transition count matrix with

$$c_{t+1,z_t,z_{t+1}} = c_{t,z_t,z_{t+1}} + 1, \quad (25)$$

$$a_{t+1,z_t,z_{t+1}} = c_{t+1,z_t,z_{t+1}} / \sum_{k=1}^K c_{t+1,z_t,k}. \quad (26)$$

To update the state duration probabilities, we keep a count of the duration steps s_t in which the cluster assignment is the same, i.e.,

$$s_{t+1} = \begin{cases} s_t + 1, & \text{if } z_{t+1} = z_t, \\ 0, & \text{otherwise.} \end{cases} \quad (27)$$

Let us denote n_{t,z_t} as the total number of transitions to other states from the state z_t till time t . When the subsequent cluster assignment is different, $z_{t+1} \neq z_t$, the duration count is reset to zero, $s_{t+1} = 0$, the transition count to other states is incremented, $n_{t+1,z_t} = n_{t,z_t} + 1$, and the duration model

[¶]Setting $d_i = 0$ by choosing $\lambda_1 \gg 0$ gives the loss function formulation with isotropic Gaussian under small variance asymptotics (Roychowdhury et al. 2013).

parameters $\{\mu_{t+1,z_t}^S, \Sigma_{t+1,z_t}^S\}$ are updated as

$$\mu_{t+1,z_t}^S = \mu_{t,z_t}^S + \frac{(s_t - \mu_{t,z_t}^S)}{n_{t,z_t} + 1}, \quad (28)$$

$$e_{t+1,z_t} = e_{t,z_t} + (s_t - \mu_{t,z_t}^S)(s_t - \mu_{t+1,z_t}^S), \quad (29)$$

$$\Sigma_{t+1,z_t}^S = \frac{e_{t+1,z_t}}{n_{t,z_t}}. \quad (30)$$

The parameters updated at time step $t + 1$ minimizes the loss function

$$\begin{aligned} \mathcal{L}(z_{t+1}, d_{t+1,z_{t+1}}, \mu_{t+1,z_{t+1}}, U_{t+1,z_{t+1}}^{d_{t+1,z_{t+1}}}, a_{t+1,z_{t+1}}) = \\ \lambda(K-1) + \lambda_1 d_{t+1,z_{t+1}} - \lambda_2 \log(a_{z_t,z_{t+1}}) + \lambda_3 \tau_{z_{t+1}} \\ + \text{dist}(\xi_{t+1}, \mu_{t+1,z_{t+1}}, U_{t+1,z_{t+1}}^{d_{t+1,z_{t+1}}})^2 \\ \leq \mathcal{L}(z_{t+1}, d_{t,z_{t+1}}, \mu_{t,z_{t+1}}, U_{t,z_{t+1}}^{d_{t,z_{t+1}}}, a_{t,z_{t+1}}). \end{aligned}$$

A decrease of the loss function ensures that the assigned cluster parameters are updated in an optimal manner. In case a new cluster is assigned to the datapoint, the loss function at time t is evaluated with the cluster having the lowest cost among the existing set of clusters.

Remark: Note that λ_2 encourages visiting the more influential states, and λ_3 restricts creation of new states. We do not explicitly penalize the deviation from the state duration distribution in the cluster assignment step or the loss function, and only re-estimate the parameters of the state duration in the parameter update step. Deviation from the state duration parameters may also be explicitly penalized as shown with small variance asymptotic analysis of hidden Markov jump processes (Huggins et al. 2015).

6 Scalable Online Sequence Clustering (SOSC)

SOSC is an unsupervised non-parametric online learning algorithm for clustering time-series data. It incrementally projects the streaming data in its low dimensional subspace and maintains a history of the duration steps and the subsequent transition to other subspaces. The projection mechanism uses a non-parametric locally linear principal component analysis whose redundant dimensions are automatically discarded by small variance asymptotic analysis along those dimensions, while the spatio-temporal information is stored with an infinite state hidden semi-Markov model. During learning, if a cluster evolves such that it is closer to another cluster than the threshold λ , the two clusters are merged into one and the subspace of the dominant cluster is retained. The overall algorithm is shown in Alg. 1 (see Extension A-3 for associated codes and examples).

The algorithm yields a generative model that scales well in higher dimensions and does not require computation of numerically unstable gradients for the parameter updates at each iteration. These desirable aspects of the model comes at a cost of hard/deterministic clusters which could be a bottleneck for some applications. Non-parametric treatment aids the user to build the model online without specifying the number of clusters and the subspace dimension of each cluster, as the parameter set grows with the size/complexity of the data during learning. The

Algorithm 1 Scalable Online Sequence Clustering (SOSC)

Input: $\langle \lambda, \lambda_1, \lambda_2, \lambda_3, \sigma^2, b_m \rangle$

procedure SOSC

```

1: Initialize  $K := 1, \{d_{0,K}, c_{0,K,K}, \mu_{0,K}^S, n_{0,K}, e_K\} := 0$ 
2: while new  $\xi_{t+1}$  is added do
3:    $z_{t+1} = \arg \min_{i=1:K+1} \begin{cases} q_{1,i}, & \text{if } \{a_{t,z_t,i} > 0, i \leq K\} \\ q_{2,i}, & \text{if } \{a_{t,z_t,i} = 0, i \leq K\} \\ q_{3,i}, & \text{otherwise,} \end{cases}$ 
   by computing  $q_{1,i}, q_{2,i}, q_{3,i}$  using Eq. (22), (23), (24)
4:   if  $z_{t+1} = K + 1$  then
5:      $K := K + 1, \mu_{t+1,K} := \xi_t, \Sigma_{t+1,K} := \sigma^2 I$ 
6:      $\{d_{t+1,K}, c_{t+1,K,K}, \mu_{old,K}^S, n_{t+1,K}, e_{t+1,K}\} := 0$ 
7:   else
8:     Update  $\mu_{t+1,z_{t+1}}$  using Eq. (10)
9:     Solve  $R_{t+1,z_{t+1}}$ , update  $U_{t+1,z_{t+1}}^{d_{t+1,z_{t+1}}}$  using Eq. (14)
10:    Update  $d_{t+1,z_{t+1}}$  using Eq. (19)
11:    Update  $\Sigma_{t+1,z_{t+1}}$  using Eq. (21)
12:   end if
13:   Update  $c_{t+1,z_{t+1}}, a_{t+1,z_{t+1}}$  using Eq. (25), (26)
14:   if  $z_{t+1} = z_t$  then
15:      $s_{t+1} := s_t + 1$ 
16:   else
17:      $s_{t+1} := 0, n_{t+1,z_t} := n_{t,z_t} + 1$ 
18:     Update  $\mu_{t+1,z_t}^S$  using Eq. (28)
19:     Update  $e_{t+1,z_t}$  using Eq. (29)
20:     Update  $\Sigma_{t+1,z_t}^S$  using Eq. (30) for  $n_{t,z_t} > 1$ 
21:   end if
22:    $z_t := z_{t+1}$ 
23:   for  $i := 1$  to  $K$  do
24:     if  $\|\mu_{t+1,z_{t+1}} - \mu_{t,i}\|_2 < \lambda$  then  $\{i \neq z_{t+1}\}$ 
25:       Merge_Clusters( $z_{t+1}, i$ )
26:     end if
27:   end for
28: end while
29: return  $\theta_{t,s}^* = \{\mu_{t,i}, \Sigma_{t,i}, \{a_{t,i,j}\}_{j=1}^K, \mu_{t,i}^S, \Sigma_{t,i}^S\}_{i=1}^K$ 
```

penalty parameters introduced are more intuitive to specify and act as regularization terms for model selection based on the structure of the data. Note that the order of the streaming data plays an important role during learning, and multiple starts from different initial configurations may lead to different solutions as we update the model parameters after registering every new sample. Alternatively, the model parameters can be initialized with a batch algorithm after storing a few demonstrations, or the parameters can be updated sequentially in a mini-batch manner. Systematic investigation of these approaches is subject to future work.

6.1 Task-Parameterized Formulation of SOSC

Task-parameterized models provide a probabilistic formulation to deal with different real world situations by adapting the model parameters in accordance with the external task parameters that describe the environment/configuration/situation, instead of hard coding the solution for each new situation or handling it in an *ad hoc* manner (Wilson and Bobick 1999; Calinon 2016; Tanwani and Calinon 2016). Task-parameterized formulation of the SOSC model is able to handle new situations by

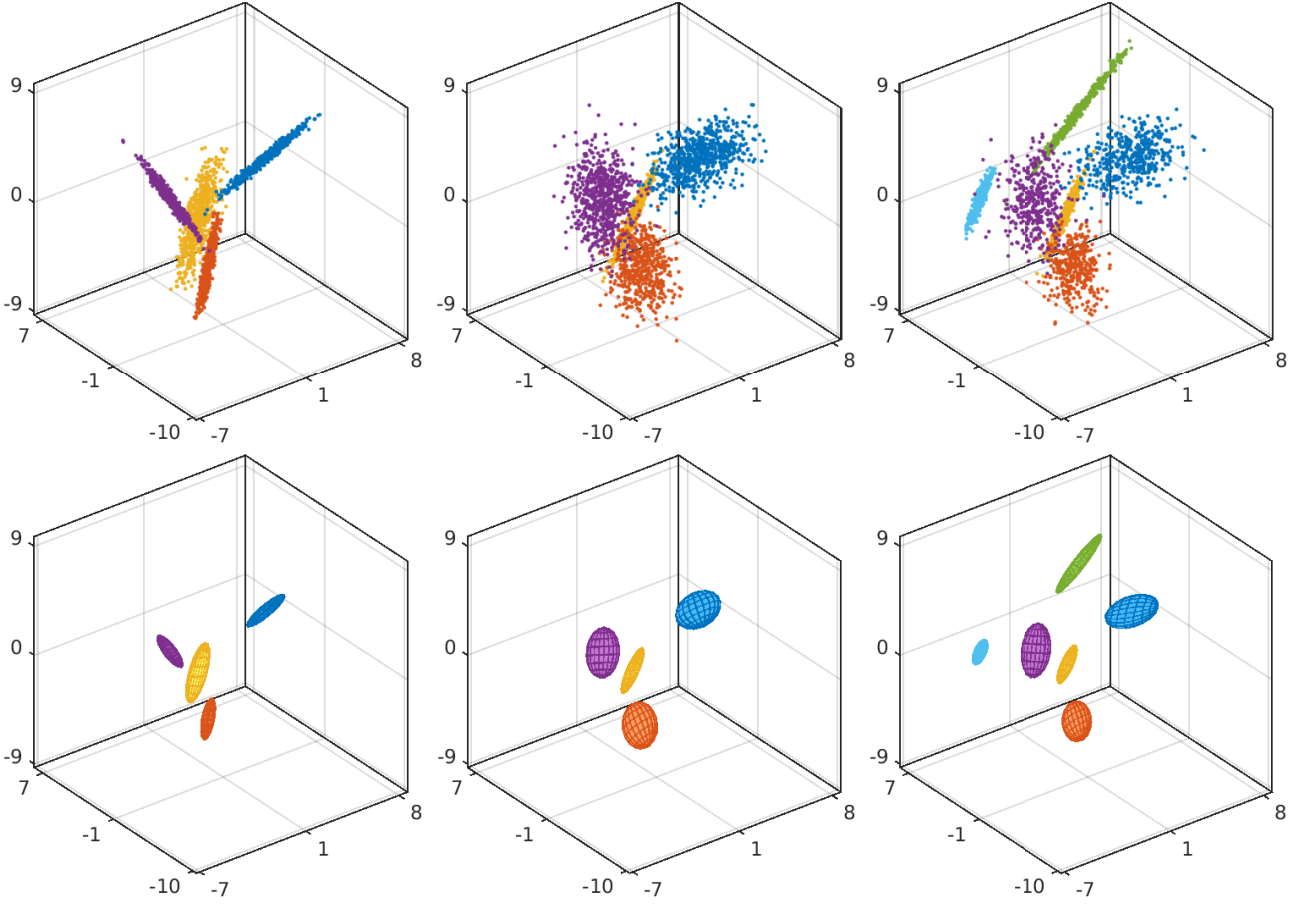


Figure 2. Non-stationary data shown on top is encoded with the SOSC model on bottom: (left) $K := 4$, d_k is randomly chosen, $t := 1 \dots 2500$, (middle) $K := 4$, $d_k := D - d_k$, $t := 2501 \dots 5000$, (right) $K := 6$, d_k is the same as before, $t := 5001 \dots 7500$.

defining external frames of reference such as a frame can be attached to an object whose position and orientation may change during the task. When a different situation occurs (position/orientation of the object changes), changes in the task parameters/reference frames are used to modulate the model parameters in order to adapt the robot movement to the new situation.

We represent the task parameters with P coordinate systems, defined by $\{\mathbf{A}_{t,j}, \mathbf{b}_{t,j}\}_{j=1}^P$, where $\mathbf{A}_{t,j}$ denotes the orientation of the frame as a rotation matrix and $\mathbf{b}_{t,j}$ represents the origin of the frame at time t . Each demonstration ξ_t is observed from the viewpoint of P different experts/frames, with $\xi_t^{(j)} = \mathbf{A}_{t,j}^{-1}(\xi_t - \mathbf{b}_{t,j})$ denoting the demonstration observed with respect to frame j . The parameters of the task-parameterized SOSC model are defined by $\theta_{t,p} = \left\{ \{\mu_{t,i}^{(j)}, \Sigma_{t,i}^{(j)}\}_{j=1}^P, \{a_{t,i,m}\}_{m=1}^K, \mu_{t,i}^S, \Sigma_{t,i}^S \right\}_{i=1}^K$, where $\mu_{t,i}^{(j)}$ and $\Sigma_{t,i}^{(j)}$ define the mean and the covariance matrix of i -th mixture component in frame j at time t . Parameter updates of the task-parameterized SOSC algorithm remain the same as described in Alg. 1, except the computation of the mean and the covariance matrix is repeated for each frame separately.

In order to combine the output of the experts for an unseen situation represented by the frames $\{\mathbf{A}_{t,j}, \mathbf{b}_{t,j}\}_{j=1}^P$, we linearly transform the Gaussians back to the global coordinates with $\{\tilde{\mathbf{A}}_{t,j}, \tilde{\mathbf{b}}_{t,j}\}_{j=1}^P$, and retrieve the new model parameters $\{\tilde{\mu}_{t,i}, \tilde{\Sigma}_{t,i}\}$ for the i -th mixture component by computing the products of the linearly transformed

Gaussians

$$\mathcal{N}(\tilde{\mu}_{t,i}, \tilde{\Sigma}_{t,i}) \propto \prod_{j=1}^P \mathcal{N}(\tilde{\mathbf{A}}_{t,j} \mu_{t,i}^{(j)} + \tilde{\mathbf{b}}_{t,j}, \tilde{\mathbf{A}}_{t,j} \Sigma_{t,i}^{(j)} \tilde{\mathbf{A}}_{t,j}^\top). \quad (31)$$

The product of Gaussians can be evaluated in a closed form solution with

$$\begin{aligned} \tilde{\mu}_{t,i} &= \tilde{\Sigma}_{t,i} \sum_{j=1}^P \left(\tilde{\mathbf{A}}_{t,j} \Sigma_{t,i}^{(j)} \tilde{\mathbf{A}}_{t,j}^\top \right)^{-1} \left(\tilde{\mathbf{A}}_{t,j} \mu_{t,i}^{(j)} + \tilde{\mathbf{b}}_{t,j} \right), \\ \tilde{\Sigma}_{t,i} &= \left(\sum_{j=1}^P \left(\tilde{\mathbf{A}}_{t,j} \Sigma_{t,i}^{(j)} \tilde{\mathbf{A}}_{t,j}^\top \right)^{-1} \right)^{-1}. \end{aligned} \quad (32)$$

Under the small variance asymptotics, the loss function at time step $t+1$ for the task-parametrized SOSC model with the resulting $\mathcal{N}(\tilde{\mu}_{t+1,z_{t+1}}, \tilde{\Sigma}_{t+1,z_{t+1}})$ yields

$$\begin{aligned} \mathcal{L}(z_{t+1}, \tilde{d}_{t+1,z_{t+1}}, \tilde{\mu}_{t+1,z_{t+1}}, \tilde{\mathbf{U}}_{t+1,z_{t+1}}^{\tilde{d}_{t+1,z_{t+1}}}, a_{t+1,z_{t+1}}) &= \\ \lambda(K-1) + \lambda_1 \tilde{d}_{t+1,z_{t+1}} - \lambda_2 \log(a_{t+1,z_{t+1}}) + \lambda_3 \tau_{z_{t+1}} &+ \text{dist}(\xi_{t+1}, \tilde{\mu}_{t+1,z_{t+1}}, \tilde{\mathbf{U}}_{t+1,z_{t+1}}^{\tilde{d}_{t+1,z_{t+1}}})^2 \\ &\leq \mathcal{L}(z_{t+1}, \tilde{d}_{t+1,z_{t+1}}, \tilde{\mu}_{t+1,z_{t+1}}, \tilde{\mathbf{U}}_{t+1,z_{t+1}}^{\tilde{d}_{t+1,z_{t+1}}}, a_{t+1,z_{t+1}}), \end{aligned}$$

where $\tilde{\mathbf{U}}_{t+1,z_{t+1}}^{\tilde{d}_{t+1,z_{t+1}}}$ corresponds to the basis vectors of the resulting $\tilde{\Sigma}_{t+1,z_{t+1}}$ and $\tilde{d}_{t+1,z_{t+1}} = \min_j d_{t+1,z_{t+1}}^{(j)}$, i.e., the

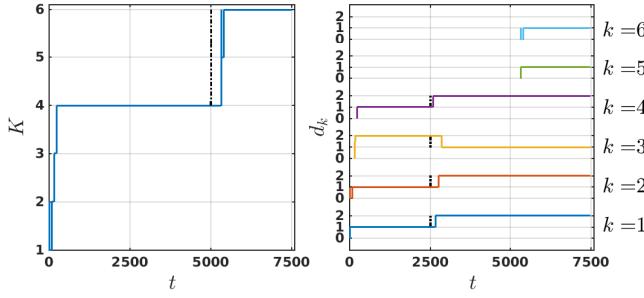


Figure 3. Evolution of K and d_k with number of instances.

product of Gaussians subspace dimension is defined by the minimum of corresponding subspace dimensions of the Gaussians in P frames for the z_{t+1} mixture component.

7 Experiments, Results and Discussion

In this section, we first evaluate the performance of the SOSOC model to encode the synthetic data with a 3-dimensional illustrative example followed by its capability to scale in high dimensional spaces. We then consider a real-world application of learning robot manipulation tasks for semi-autonomous teleoperation with the proposed task-parameterized SOSOC algorithm. The goal is to assess the performance of the SOSOC model to handle noisy online time-series data in a parsimonious manner.

7.1 Synthetic Data

7.1.1 Non-Stationary Learning with 3-Dimensional Data:

We consider a 3-dimensional stream of datapoints $\xi_t \in \mathbb{R}^3$ generated by stochastic sampling from a mixture of different clusters that are connected in a left-right cyclic HSMM. The centers of the clusters are successively drawn from the interval $[-5, 5]$ such that the next cluster is at least $4\sqrt{D}$ units farther than the existing set of clusters. Subspace dimension of each cluster is randomly chosen to lie up to $(D - 1)$ dimensions (a line or a plane for 3-dimensions), and the basis vectors are sampled randomly in that subspace. Duration steps in a given state are sampled from a uniform distribution in the interval $[70, 90]$ after which the data is subsequently generated from the next cluster in the model in a cyclic manner. A white noise of $\mathcal{N}(0, 0.04I)$ is added to each sampled data point. Model learning is divided in three stages: 1) for the first 2500 instances, the number of clusters is set to 4 and the subspace dimension of each cluster is fixed, 2) for the subsequent 2500 instances, we change the subspace dimension of each cluster to $(D - d_k)$ for $k = 1 \dots K$ (for example, a line becomes a plane), while keeping the same number of clusters, and 3) two more clusters are then added in the mixture model for the next 2500 instances without any change in the subspace dimension of the previous clusters. The parameters are defined as $\{\lambda = 3.6, \lambda_1 = 0.35, \lambda_2 = \lambda_3 = 0.025, \sigma^2 = 0.15, b_m = 50\}$. The weights of the parameter update are based on eligibility traces as in Eq. (12) with a discount factor of 0.995.

Results of the learned model are shown in Fig. 2. We can see that the SOSOC model is able to efficiently encode the number of clusters and the subspace dimension of

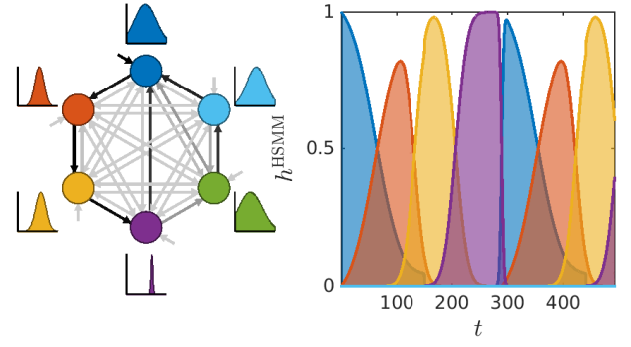


Figure 4. (left) Learned HSMM transition matrix and state duration model representation with $s^{\max} = 150$, (right) rescaled forward variable, $h_{t,t}^{\text{HSMM}} = \frac{\alpha_{t,t}^{\text{HSMM}}}{\sum_{k=1}^K \alpha_{t,k}^{\text{HSMM}}}$, sampled from initial position.

each cluster in each stage of the learning process. The model projects each datapoint in the subspace of the nearest cluster contrary to the K -means clustering based on the Euclidean distance metric only. The model is able to adapt the subspace dimension of each cluster in the second stage of the learning process and subsequently incorporate more clusters in the final stage with the non-stationary data. Fig. 3 shows the evolution of the number of clusters and the subspace dimension of each cluster with the streaming data. Note that the encoding problem is considerably hard here as the model starts with one cluster only and adapts during the learning process. Clusters that evolve to come closer than a certain threshold are merged during the learning process. Fig. 4 shows the graphical model representation of the learned HSMM with the state transitions and the state duration model, along with a sample of the forward variable generated from the initial position (see Eq. (40)).

7.1.2 Stationary Learning with High-Dimensional Data:

In this experiment, we sample the data from a stationary distribution corresponding to the first stage of the previous example where $K = 4$ and the subspace of each cluster does not change in the streaming data. Dimensionality of the data is successively chosen from the set $D = \{10, 25, 50, 75\}$, and the number of instances are varied for each dimension from the set $T = \{1000, 2500, 5000, 7500\}$. Parameter λ is experimentally selected for each dimension to achieve satisfying results and the weights of the parameter update are linearly incremented for each cluster. Fig. 5 shows the performance of the SOSOC model to encode data in high dimensions averaged over 10 iterations. Our results show that the algorithm yields a compact encoding as indicated by high values of the average *silhouette score* (SS),^{||} and the

^{||}Silhouette score (SS) measures the tightness of a cluster relative to the other clusters without using any labels,

$$SS_i \triangleq \frac{b_i - a_i}{\max\{a_i, b_i\}}, \quad SS_i \in [-1, 1],$$

where a_i is the mean distance of ξ_i to the other points in its own cluster, and b_i is the mean distance of ξ_i to the points in the closest ‘neighbouring’ cluster.

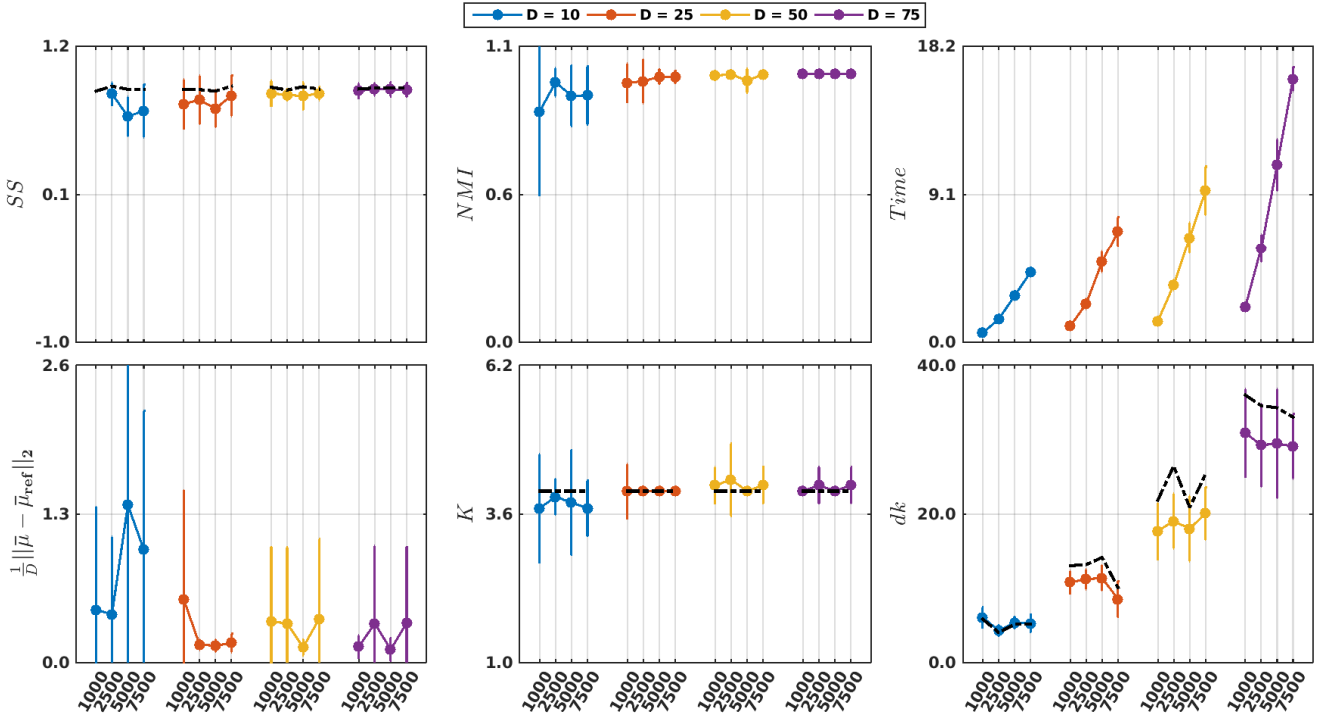


Figure 5. SOSC model evaluation to encode synthetic high-dimensional data. Results are averaged over 10 iterations. Black dotted lines indicate the reference value: (top-left) silhouette score (SS), (top-middle) normalized mutual information score (NMI), (top-right) time in seconds, (bottom-left) average distance between learned cluster means and ground truth, (bottom-middle) number of clusters, (bottom-right) average subspace dimension across all clusters.

normalized mutual information (NMI) score,** while being robust to the intrinsic subspace dimension of the data and the number of clusters.

7.2 Learning Manipulation Skills for Semi-Autonomous Teleoperation

We are interested in performing remote manipulation tasks with robots via teleoperation within the DexROV project (Gancet et al. 2015). Direct teleoperation, where the teleoperator actions are directly reproduced on the remote robot, is often infeasible due to the presence of communication latencies and noise in the feedback. Predicting/correcting the response of the operator can assist the teleoperator in executing these manipulation tasks (Dragan and Srinivasa 2013; Maeda et al. 2015). In this paper, we build the task-parameterized SOSC model online from the teleoperator demonstrations and provide a probabilistic formulation to predict his/her intention while performing the task. The model is used to recognize the intention of the teleoperator, and synthesize motion on the remote end to perform manipulation tasks in a semi-autonomous manner. Two didactic examples of manipulation tasks are incrementally learned for guided assistance: target tracking with a screwdriver and hooking a carabiner.

7.2.1 Experimental Setup: In our experimental set-up with the Baxter robot, the operator controls/teleoperates the right arm, with the left arm used as the input device. The tool (screwdriver/carabiner) is mounted on the end-effector of the right arm and the target (movable object/rotatable rod) is placed at a reachable location from the arm. Demonstrations are performed in the direct control mode where the desired pose of the right arm is computed by adding an offset

in the lateral direction to the end-effector position of the teleoperator’s arm. The teleoperator guides the tool to different target locations of the tool by visual feedback, as shown in Fig. 6.

7.2.2 Learning Problem: Let us denote $\xi_t = [\xi_t^x \ \xi_t^o]^\top$ with $\xi_t^x \in \mathbb{R}^7$ and $\xi_t^o \in \mathbb{R}^7$ representing respectively the input state of the teleoperator arm and the input state of the teleoperator arm observed in the frame of reference of the target pose of the tool (screwdriver/carabiner). The state of the teleoperator arm is represented by the position $x_t^p \in \mathbb{R}^3$ and the orientation $\varepsilon_t^o \in \mathbb{R}^4$ of the teleoperator arm end-effector in their respective frames of reference with $D = 14$. We attach a frame $\{A_{t,1}, b_{t,1}\}$ to the target pose of the tool, described by

$$A_{t,1} = \begin{bmatrix} I^x & 0 & 0 \\ 0 & R_{t,1}^o & 0 \\ 0 & 0 & \mathcal{E}_{t,1}^o \end{bmatrix}, b_{t,1} = \begin{bmatrix} 0 \\ p_{t,1}^o \\ 0 \end{bmatrix}, \quad (33)$$

where $p_{t,1}^o \in \mathbb{R}^3$, $R_{t,1}^o \in \mathbb{R}^{3 \times 3}$, $\mathcal{E}_{t,1}^o \in \mathbb{R}^{4 \times 4}$ denote the Cartesian position, the rotation matrix and the quaternion matrix of the frame/tool at time t respectively, see e.g., Silverio et al. (2015). Note that the frame has two

**Normalized mutual information (NMI) is an extrinsic information-theoretic measure to evaluate the alignment between the assigned cluster labels \mathcal{Z} and the ground truth cluster labels \mathcal{X} ,

$$\text{NMI}(\mathcal{Z}, \mathcal{X}) \triangleq \frac{I(\mathcal{Z}, \mathcal{X})}{[H(\mathcal{Z}) + H(\mathcal{X})]/2}, \quad \text{NMI}(\mathcal{Z}, \mathcal{X}) \in [0, 1],$$

where $I(\mathcal{Z}, \mathcal{X})$ is the mutual information and $H(\mathcal{X})$ is the entropy of cluster labels \mathcal{X} .

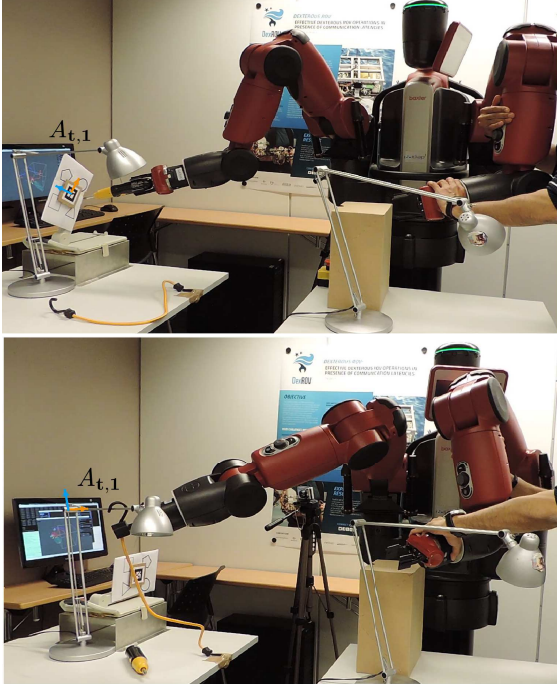


Figure 6. Semi-autonomous teleoperation with the Baxter robot for guided assistance of manipulation tools: (top) screwdriving with a frame attached to the movable target, (bottom) hooking a carabiner with a frame attached to a rotatable rod.

components, the input component represents the teleoperator pose in the global frame corresponding to ξ_t^x , while the output component maps the teleoperator state with respect to the target pose corresponding to ξ_t^o . The observation variable ξ_t is augmented to couple the movement of the robot arm and the target, i.e., we learn the mapping between the teleoperator pose and the teleoperator pose observed in the frame of reference of the target as a joint distribution. The coupling strength depends upon the variations observed in the demonstrations. Parts of the movement with invariant characteristics appear when approaching the target where the teleoperator movement is synchronized with the target. Based on the learned joint distribution of the task-parameterized SOSC model, we seek to recognize the intention of the teleoperator and subsequently correct the current state of the teleoperated arm. In case of communication disruptions, we solicit the model to generate movement on the remote arm in an autonomous manner until further communication is re-established. We present two formulations of the algorithm to assist the teleoperator in performing remote manipulation tasks: 1) *time-independent shared control*, 2) *time-dependent autonomous control*.

Time-Independent Shared Control: We seek to leverage upon the SOSC model to adjust the movement of the robot in following the teleoperator state in a time-independent manner based on the principle of shared control. Given the current state of the teleoperator arm ξ_t^x and the task-parametrized SOSC model encoding the joint distribution as $\mathcal{N}(\tilde{\mu}_{t,i}, \tilde{\Sigma}_{t,i})$, the conditional probability distribution of the teleoperator arm with respect to the target $\mathcal{P}(\xi_t^o | \xi_t^x)$ can be approximated as $\mathcal{N}(\tilde{\mu}_t^o, \tilde{\Sigma}_t^o)$ using Gaussian mixture

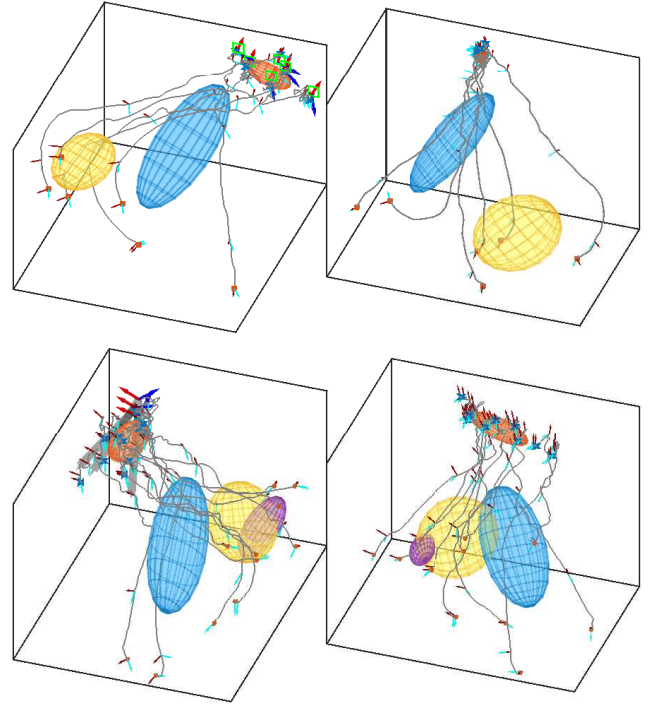


Figure 7. Joint distribution of the task-parameterized SOSC model for guided assistance in the screwdriving task (top) and hooking a carabiner task (bottom). Demonstrations and model with respect to the input part of the frame on (left), and with respect to the output part of the frame on (right).

regression (Ghahramani and Jordan 1994), namely

$$\tilde{\mu}_t^o = \sum_{i=1}^K h_i(\xi_t^x) \hat{\mu}_{t,i}^o(\xi_t^x), \quad (34)$$

$$\tilde{\Sigma}_t^o = \sum_{i=1}^K h_i(\xi_t^x) \left(\hat{\Sigma}_{t,i}^o + \hat{u}_{t,i}^o(\xi_t^x) (\hat{\mu}_{t,i}^o(\xi_t^x))^T \right) - \tilde{\mu}_t^o \tilde{\mu}_t^{oT}, \quad (35)$$

$$\text{with } h_i(\xi_t^x) = \frac{\pi_i \mathcal{N}(\xi_t^x | \tilde{\mu}_{t,i}^x, \tilde{\Sigma}_{t,i}^x)}{\sum_k \pi_k \mathcal{N}(\xi_t^x | \tilde{\mu}_{t,k}^x, \tilde{\Sigma}_{t,k}^x)}, \quad (36)$$

$$\hat{\mu}_{t,i}^o(\xi_t^x) = \tilde{\mu}_{t,i}^o + \tilde{\Sigma}_{t,i}^{oI} \tilde{\Sigma}_{t,i}^{xI-1} (\xi_t^x - \tilde{\mu}_{t,i}^x), \quad (37)$$

$$\hat{\Sigma}_{t,i}^o = \tilde{\Sigma}_{t,i}^o - \tilde{\Sigma}_{t,i}^{oI} \tilde{\Sigma}_{t,i}^{xI-1} \tilde{\Sigma}_{t,i}^{oI}, \quad (38)$$

$$\text{where } \tilde{\mu}_{t,i} = \begin{bmatrix} \tilde{\mu}_{t,i}^x \\ \tilde{\mu}_{t,i}^o \end{bmatrix}, \text{ and } \tilde{\Sigma}_{t,i} = \begin{bmatrix} \tilde{\Sigma}_{t,i}^x & \tilde{\Sigma}_{t,i}^{xo} \\ \tilde{\Sigma}_{t,i}^{ox} & \tilde{\Sigma}_{t,i}^o \end{bmatrix}.$$

The output Gaussian $\mathcal{N}(\tilde{\mu}_t^o, \tilde{\Sigma}_t^o)$ predicts the teleoperator state and the uncertainty associated with the state in the frame of reference of the target. Let us denote $\kappa^2 \mathbf{I}$ as the uncertainty associated with the teleoperator input state ξ_t^x in performing the manipulation task. Higher values of κ^2 are used when the confidence of the teleoperator in performing the task is low, for example, when the feedback is noisy, the task is complex or the teleoperator is novice, and vice versa for lower values of κ^2 . Hence, the Gaussian $\mathcal{N}(\xi_t^x, \kappa^2 \mathbf{I})$ represents the uncertainty associated with the current teleoperator state, while the Gaussian $\mathcal{N}(\tilde{\mu}_t^o, \tilde{\Sigma}_t^o)$ predicts the teleoperator state based on the invariant patterns observed in the demonstrations with respect to the target. The

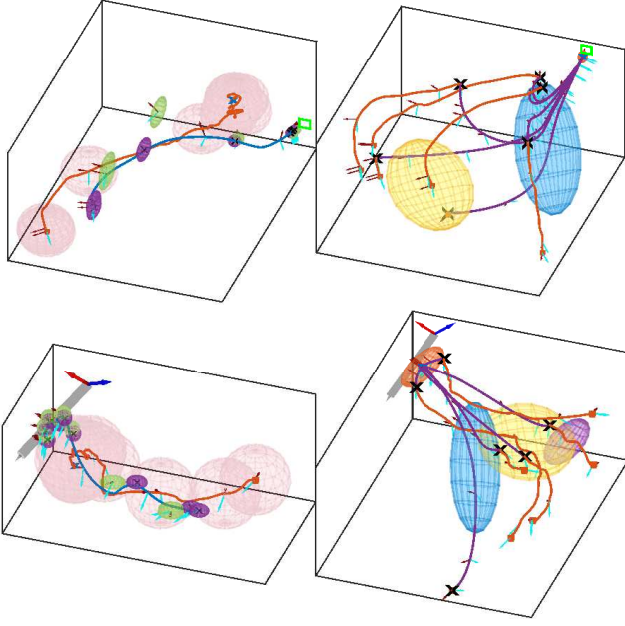


Figure 8. Semi-autonomous teleoperation for a new target pose with a screwdriver (*top*) and a carabiner (*bottom*). Shared control example (*left*): the teleoperator demonstration (in red) strays away from the target pose, while the corrected trajectory (in blue) reaches the target pose. Desired state $\mathcal{N}(\hat{\mu}_t, \hat{\Sigma}_t)$ is shown in purple, teleoperator state $\mathcal{N}(\xi_t^x, \kappa^2 \mathbf{I})$ in red, and predicted state $\mathcal{N}(\tilde{\mu}_t^o, \tilde{\Sigma}_t^o)$ in green (see Sec. 7.2.2 for details). Autonomous control example (*right*): the arm movement is randomly switched (marked with a cross) from direct control (in red) to autonomous control (in purple) in which the learned model is used to generate the movement to the target pose.

resulting desired state $\mathcal{N}(\hat{\mu}_t, \hat{\Sigma}_t)$ is obtained by taking the product of Gaussians corresponding to the teleoperator state and the predicted state, namely

$$\mathcal{N}(\hat{\mu}_t, \hat{\Sigma}_t) \propto \mathcal{N}(\xi_t^x, \kappa^2 \mathbf{I}) \mathcal{N}(\tilde{\mu}_t^o, \tilde{\Sigma}_t^o). \quad (39)$$

The resulting desired state is followed in a smooth manner with an infinite horizon linear quadratic regulator (Borrelli et al. 2011) (see Appx. C).

Time-Dependent Autonomous Control: In case of communication disruptions where it is difficult to retrieve the state of the teleoperator, the manipulation task can be completed in an autonomous manner. Task-parameterized SOSC model is used to generate the robot movement in an autonomous manner with the help of the forward variable $\alpha_{t,i}^{\text{HSMM}} \triangleq P(z_t = i, \xi_1 \dots \xi_t | \theta_h)$. Given the model parameters θ_h and the partial observation sequence $\xi_1 \dots \xi_t$, the probability of a datapoint ξ_t to be in state i at time t is recursively computed in the explicit duration HSMM using the forward variable as

$$\alpha_{t,i}^{\text{HSMM}} = \sum_{j=1}^K \min(s^{\max}, t-1) \sum_{s=1}^{\alpha_{t-s,j}^{\text{HSMM}}} a_{j,i} \mathcal{N}(s | \mu_i^S, \Sigma_i^S) \prod_{c=t-s+1}^t \mathcal{N}(\xi_c | \tilde{\mu}_i, \tilde{\Sigma}_i). \quad (40)$$

The forward variable is used to evaluate the current state of the task ξ_{t_o} using $\alpha_{t_o,i}^{\text{HSMM}} = \frac{\pi_i \mathcal{N}(\xi_{t_o} | \tilde{\mu}_i, \tilde{\Sigma}_i)}{\sum_{k=1}^K \pi_k \mathcal{N}(\xi_{t_o} | \tilde{\mu}_k, \tilde{\Sigma}_k)}$, and

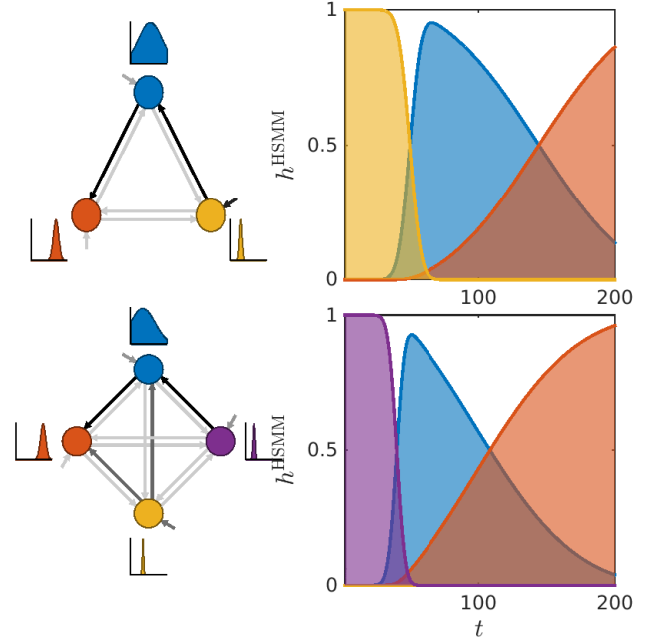


Figure 9. HSMM graphical model representation ($s^{\max} = 150$) along with evolution of the rescaled forward variable for screwdriving (*top*) and hooking a carabiner (*bottom*).

subsequently plan the movement sequence for the next T steps with $t = (t_o + 1) \dots T$. Note that only the transition matrix and the duration model are used to plan the future evolution of the initial/current state ξ_{t_o} (the influence of the spatial data is omitted as it has not been observed), i.e., $\mathcal{N}(\xi_t | \tilde{\mu}_i, \tilde{\Sigma}_i) = 1$ for $t = (t_o + 1) \dots T$. This is used to retrieve a stepwise reference trajectory $\mathcal{N}(\hat{\mu}_t, \hat{\Sigma}_t)$ from the state sequence z_t computed from the forward variable. The stepwise reference trajectory is tracked in a smooth manner with a finite-horizon linear quadratic tracking controller (Tanwani and Calinon 2016) (see Appx. D), based on

$$z_t = \arg \max_i \alpha_{t,i}^{\text{HSMM}}, \quad \hat{\mu}_t = \tilde{\mu}_{z_t}^o, \quad \hat{\Sigma}_t = \tilde{\Sigma}_{z_t}^o. \quad (41)$$

We exploit the time-dependent autonomous control formulation to assist the teleoperator in performing challenging tasks and/or to counter large communication latencies. The teleoperator switches at any point in time t_o to the autonomous mode upon which the robot arm successively re-plans and executes the task for the next T steps. When the task is accomplished or the communication channel is re-established, the operator may switch back to the manual control upon which the robot arm returns to the teleoperated state.

7.2.3 Results and Discussions: We collect 6 kinesthetic demonstrations for screwdriving with the initial pose of the target rotated/translated in the successive demonstrations, and perform 11 demonstrations of hooking a carabiner at various places on the rod for 3 different rotated configurations of the rod segment. Demonstrations are subsampled around 7 Hz and limited to 200 datapoints for each demonstration. The parameters are defined as $\{\lambda = 0.65, \lambda_1 = 0.03, \lambda_2 = 0.001, \lambda_3 = 0.04, \sigma^2 = 2.5 \times 10^{-4}, \kappa^2 = 0.01\}$.

Table 1. Performance comparison of the SOSC model against parametric batch HSMM models using number of parameters N_p , and the endpoint error between the teleoperated arm and the target. Teleoperation modes are direct control (DC), shared control (SC) and autonomous control (AC). Errors are meters.

Model	N_p	DC Error	SC Error	AC Error
screw-driving task ($K = 3, D = 14$)				
FC-HSMM	372	0.30± 0.17	0.095± 0.025	0.038± 2.5×10^{-5}
ST-HSMM	295		0.094± 0.026	0.037± 1.8×10^{-5}
MFA-HSMM ($d_k = 4$)	267		0.099± 0.022	0.037 ± 7.7×10^{-6}
SOSC ($\bar{d}_k = 3.67$)	211		0.084 ± 0.018	0.043± 1.3×10^{-4}
hooking carabiner task ($K = 4, D = 14$)				
FC-HSMM	500	0.10± 0.062	0.081± 0.056	0.099± 0.068
ST-HSMM	332		0.082± 0.058	0.022 ± 2.6×10^{-4}
MFA-HSMM ($d_k = 4$)	360		0.08 ± 0.056	0.037± 8.8×10^{-4}
SOSC ($\bar{d}_k = 4.25$)	318		0.08 ± 0.056	0.073± 3.7×10^{-4}

Results of the task-parameterized SOSC model for the two tasks are shown in Fig. 7. We observe that the model exploits the variability in the demonstrations to statistically encode different phases of the task in the joint distribution. Demonstrations corresponding to the input component of the frame encode the reaching movement to different target poses with the screwdriver and the carabiner in the global frame, while the output component of the frame represents this movement observed from the viewpoint of the target (respectively shown as converging to a point for the screwdriver and to a line for the carabiner). The learned model for the screwdriving task contains 3 clusters with subspace dimensions $\{4, 3, 4\}$, while the carabiner task model contains 4 clusters with subspace dimensions $\{5, 5, 4, 3\}$.

Fig. 8 (left) shows how the model adjusts the movement of the teleoperator based on his/her current state in a time-independent manner. When the teleoperator is away from the target, the variance in the output distribution $\mathcal{N}(\tilde{\mu}_t^\circ, \tilde{\Sigma}_t^\circ)$ is high and its product with the teleoperator frame $\mathcal{N}(\xi_t^x, \kappa^2 \mathbf{I})$ yields the desired state $\mathcal{N}(\tilde{\mu}_t, \tilde{\Sigma}_t)$ closer to the teleoperator as in direct teleoperation. As the teleoperator moves closer to the target and visits low variance segments of $\mathcal{N}(\tilde{\mu}_t^\circ, \tilde{\Sigma}_t^\circ)$, the desired state moves closer to the target as compared to the teleoperator. Consequently, the shared control formulation corrects the movement of the teleoperator when the teleoperator is straying from the target. Table 1 shows the performance improvement of shared control over direct control where the endpoint error is reduced from 0.3 to 0.084 meters for the screwdriving task, and from 0.1 to 0.08 meters for the carabiner task. Error is measured at the end of the demonstration from the end-effector of the teleoperated arm to the target of the screwdriver, and to the

rod segment for hooking the carabiner (see Extension A-1 for semi-autonomous teleoperation experiments and results).

To evaluate the autonomous control mode of the task-parameterized SOSC model, the teleoperator performs 6 demonstrations and switches to the autonomous mode randomly while performing the task. The teleoperated arm evaluates the current state of the task and generates the desired sequence of states to be visited for the next T steps using the forward variable as shown in Fig. 9. Fig. 8 (right) shows that the movement of the robot converges to the target from different initial configurations of the teleoperator. The obtained results are repeatable and more precise than the direct and the shared control results as shown in Table 1. Moreover, the table also compares the performance of the SOSC algorithm against several parametric batch versions of HSMMs with different covariance models in the output state distribution, including full covariance (FC-HSMM), semi-tied covariance (ST-HSMM), and MFA decomposition of covariance (MFA-HSMM). Results of the SOSC model are used as a benchmark for model selection of the batch algorithms. We can see that the proposed non-parametric online learning model gives comparable performance to other parametric batch algorithms with a more parsimonious representation (reduced number of model parameters).

In our future work, we plan to bootstrap the online learning process with the batch algorithm after a few initial demonstrations of the task. We would like to use the initialized model to make an educated guess about the penalty parameters for non-parametric online learning. We seek to extend the framework for learning a set of robot manipulation tasks as the teleoperator performs each task with a few demonstrations. Moreover, it would be interesting to test the model under more realistic environments with large communication latencies typically observed in satellite communication.

8 Conclusions

Non-parametric online learning is a promising way to adapt the model on the fly with new training data. In this paper, we have presented a non-parametric scalable online sequence clustering algorithm by online inference in DP-MPPCA and HDP-HSMM under small variance asymptotics. The algorithm incrementally clusters the streaming data with non-parametric locally linear principal component analysis and encodes the spatio-temporal patterns using an infinite hidden semi-Markov model. Non-parametric treatment gives the flexibility to continuously adapt the model with new incoming data. Learning the model online from a few human demonstrations is a pragmatic approach to teach new skills to robots. The proposed skill encoding scheme is potentially applicable to a wide range of tasks, while being robust to varying environmental conditions with the task-parameterized formulation. We show the efficacy of the approach to learn manipulation tasks online for semi-autonomous teleoperation, and assist the operator with shared control and/or autonomous control when performing remote manipulation tasks.

Funding

This work was in part supported by the DexROV project through the EC Horizon 2020 programme (Grant #635491).

References

- Argall BD, Chernova S, Veloso M and Browning B (2009) A survey of robot learning from demonstration. *Robot. Auton. Syst.* 57(5): 469–483.
- Asfour T, Azad P, Gyarfas F and Dillmann R (2008) Imitation learning of dual-arm manipulation tasks in humanoid robots. *I. J. Humanoid Robotics* 5(2): 183–202.
- Beal MJ, Ghahramani Z and Rasmussen CE (2002) The infinite hidden markov model. In: *Machine Learning*. pp. 29–245.
- Bellas A, Bouveyron C, Cottrell M and Lacaille J (2013) Model-based clustering of high-dimensional data streams with online mixture of probabilistic pca. *Advances in Data Analysis and Classification* 7(3): 281–300.
- Billard AG, Calinon S and Dillmann R (2016) Learning from humans. In: Siciliano B and Khatib O (eds.) *Handbook of Robotics*, chapter 74. Secaucus, NJ, USA: Springer, pp. 1995–2014. 2nd Edition.
- Borrelli F, Bemporad A and Morari M (2011) *Predictive control for linear and hybrid systems*. Cambridge University Press.
- Bouveyron C and Brunet C (2014) Model-based clustering of high-dimensional data: A review. *Computational Statistics and Data Analysis* 71: 52–78.
- Broderick T, Kulis B and Jordan MI (2013) Mad-bayes: Map-based asymptotic derivations from bayes. In: *Proceedings of the 30th International Conference on Machine Learning, ICML 2013, Atlanta, GA, USA, 16-21 June 2013*. pp. 226–234.
- Bruno D, Calinon S and Caldwell DG (2016) Learning autonomous behaviours for the body of a flexible surgical robot. *Autonomous Robots* : 1–15 DOI:10.1007/s10514-016-9544-6.
- Calinon S (2016) A tutorial on task-parameterized movement learning and retrieval. *Intelligent Service Robotics* 9(1): 1–29. DOI:10.1007/s11370-015-0187-9.
- Calinon S, D’halluin F, Sauser EL, Caldwell DG and Billard AG (2010) Learning and reproduction of gestures by imitation: An approach based on hidden Markov model and Gaussian mixture regression. *IEEE Robotics and Automation Magazine* 17(2): 44–54.
- Campbell T, Liu M, Kulis B, How JP and Carin L (2013) Dynamic clustering via asymptotics of the dependent dirichlet process mixture. In: Burges CJC, Bottou L, Ghahramani Z and Weinberger KQ (eds.) *NIPS*. pp. 449–457.
- Chen M, Silva JG, Paisley JW, Wang C, Dunson DB and Carin L (2010) Compressive sensing on manifolds using a nonparametric mixture of factor analyzers: Algorithm and performance bounds. *IEEE Trans. Signal Processing* 58(12): 6140–6155.
- Dempster AP, Laird NM and Rubin DB (1977) Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society B* 39(1): 1–38.
- Dragan AD and Srinivasa SS (2013) A policy-blending formalism for shared control. *I. J. Robotic Res.* 32(7): 790–805.
- Fox EB, Sudderth EB, Jordan MI and Willsky AS (2008) An hdp-hmm for systems with state persistence. In: *Proceedings of the 25th International Conference on Machine Learning, ICML ’08*. pp. 312–319.
- Gancet J, Urbina D, Letier P, Ilzokvitz M, Weiss P, Gauch F, Antonelli G, Indiveri G, Casalino G, Birk A, Pfingsthorn MF, Calinon S, Tanwani A, Turetta A, Walen C and Guilpain L (2015) Dexrov: Dexterous undersea inspection and maintenance in presence of communication latencies. *IFAC-PapersOnLine* 48(2): 218 – 223. DOI:http://dx.doi.org/10.1016/j.ifacol.2015.06.036.
- Ghahramani Z and Jordan MI (1994) Supervised learning from incomplete data via an EM approach. In: Cowan JD, Tesauro G and Alspector J (eds.) *Advances in Neural Information Processing Systems*, volume 6. San Francisco, CA, USA: Morgan Kaufmann Publishers, Inc., pp. 120–127.
- Gijsberts A and Metta G (2013) Real-time model learning using incremental sparse spectrum gaussian process regression. *Neural Networks* 41: 59 – 69. Special Issue on Autonomous Learning.
- Hoyos J, Prieto F, Alenyà G and Torras C (2016) Incremental learning of skills in a task-parameterized gaussian mixture model. *Journal of Intelligent and Robotic Systems* 82(1): 81–99.
- Huggins JH, Narasimhan K, Saeedi A and Mansinghka VK (2015) Jump-means: Small-variance asymptotics for markov jump processes. In: *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*. pp. 693–701.
- Jiang K, Kulis B and Jordan MI (2012) Small-variance asymptotics for exponential family dirichlet process mixture models. In: Pereira F, Burges CJC, Bottou L and Weinberger KQ (eds.) *Advances in Neural Information Processing Systems* 25. Curran Associates, Inc., pp. 3158–3166.
- Johnson MJ and Willsky AS (2013) Bayesian nonparametric hidden semi-markov models. *J. Mach. Learn. Res.* 14(1): 673–701.
- Krishnan S, Garg A, Patil S, Lea C, Hager G, Abbeel P and Goldberg K (2015) Transition state clustering: Unsupervised surgical trajectory segmentation for robot learning. In: *Proc. Intl Symp. on Robotics Research (ISRR)*.
- Kronander K, Khansari M and Billard A (2015) Incremental motion learning with locally modulated dynamical systems. *Robotics and Autonomous Systems* 70: 52–62.
- Kulic D, Takano W and Nakamura Y (2008) Incremental learning, clustering and hierarchy formation of whole body motion patterns using adaptive hidden markov chains. *Intl Journal of Robotics Research* 27(7): 761–784.
- Kulis B and Jordan MI (2012) Revisiting k-means: New algorithms via bayesian nonparametrics. In: *Proceedings of the 29th International Conference on Machine Learning (ICML-12)*. New York, NY, USA: ACM, pp. 513–520.
- Lee D and Ott C (2010) Incremental motion primitive learning by physical coaching using impedance control. In: *Proc. IEEE/RSJ Intl Conf. on Intelligent Robots and Systems (IROS)*. Taipei, Taiwan, pp. 4133–4140.
- Lee D, Ott C and Nakamura Y (2010) Mimetic communication model with compliant physical contact in human - humanoid interaction. *I. J. Robotic Res.* 29(13): 1684–1704.
- Maeda G, Neumann G, Ewerton M, Lioutikov R and Peters J (2015) A probabilistic framework for semi-autonomous robots based on interaction primitives with phase estimation. In: *International Symposium of Robotics Research*.
- McLachlan GJ, Peel D and Bean RW (2003) Modelling high-dimensional data by mixtures of factor analyzers.

- Computational Statistics and Data Analysis* 41(3-4): 379–388.
- Neal RM and Hinton GE (1999) A view of the EM algorithm that justifies incremental, sparse, and other variants. In: *Learning in graphical models*. Cambridge, MA, USA: MIT Press, pp. 355–368.
- Nguyen-Tuong D, Seeger M and Peters J (2009) Model learning with local gaussian process regression. *Advanced Robotics* 23(15): 2015–2034.
- Niekum S, Osentoski S, Konidaris G and Barto AG (2012) Learning and generalization of complex tasks from unstructured demonstrations. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems*. pp. 5239–5246.
- Pitman J (2002) Poisson-dirichlet and gem invariant distributions for split-and-merge transformations of an interval partition. *Combinatorics, Probability and Computing* 11: 501–514.
- Rabiner LR (1989) A tutorial on hidden Markov models and selected applications in speech recognition. *Proc. IEEE* 77:2: 257–285.
- Roychowdhury A, Jiang K and Kulis B (2013) Small-variance asymptotics for hidden markov models. In: *Advances in Neural Information Processing Systems* 26. Curran Associates, Inc., pp. 2103–2111.
- Schaal S, Ijspeert A and Billard A (2003) Computational approaches to motor learning by imitation. *Philosophical Transaction of the Royal Society of London: Series B, Biological Sciences* 358(1431): 537–547.
- Schaal S, Mohajerian P and Ijspeert A (2007) Dynamics systems vs. optimal control a unifying view. *Progress in Brain Research* 165: 425–445.
- Sethuraman J (1994) A constructive definition of Dirichlet priors. *Statistica Sinica* 4: 639–650.
- Silverio J, Roza L, Calinon S and Caldwell DG (2015) Learning bimanual end-effector poses from demonstrations using task-parameterized dynamical systems. In: *Proc. IEEE/RSJ Intl Conf. on Intelligent Robots and Systems (IROS)*. pp. 464–470.
- Song M and Wang H (2005) Highly efficient incremental estimation of Gaussian mixture models for online data stream clustering. In: *Proc. of SPIE: Intelligent Computing - Theory and Applications III*, volume 5803. pp. 174–183.
- Stulp F and Sigaud O (2015) Many regression algorithms, one unified model - A review. *Neural Networks* : 28.
- Sutton RS and Barto AG (1998) *Reinforcement Learning: An Introduction*. Cambridge, MA: MIT Press.
- Tanwani A and Calinon S (2016) Learning robot manipulation tasks with task-parameterized semitied hidden semi-markov model. *Robotics and Automation Letters, IEEE* 1(1): 235–242. DOI: 10.1109/LRA.2016.2517825.
- Teh YW, Jordan MI, Beal MJ and Blei DM (2006) Hierarchical dirichlet processes. *Journal of the American Statistical Association* 101(476): 1566–1581.
- Tipping ME and Bishop CM (1999) Mixtures of probabilistic principal component analyzers. *Neural Computation* 11(2): 443–482.
- Vakanski A, Mantegh I, Irish A and Janabi-Sharifi F (2012) Trajectory learning for robot programming by demonstration using hidden markov model and dynamic time warping. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)* 42(4): 1039–1052.
- Van Gael J, Saatchi Y, Teh YW and Ghahramani Z (2008) Beam sampling for the infinite hidden markov model. In: *Proceedings of the 25th International Conference on Machine Learning, ICML '08*. New York, NY, USA, pp. 1088–1095.
- Vijayakumar S, D'souza A and Schaal S (2005) Incremental online learning in high dimensions. *Neural Computation* 17(12): 2602–2634.
- Wang Y and Zhu J (2015) DP-space: Bayesian nonparametric subspace clustering with small-variance asymptotics. In: *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*. pp. 862–870.
- Wilson AD and Bobick AF (1999) Parametric hidden Markov models for gesture recognition. *IEEE Trans. on Pattern Analysis and Machine Intelligence* 21(9): 884–900.
- Yu SZ (2010) Hidden semi-Markov models. *Artificial Intelligence* 174: 215–243.
- Zhang Z, Chan KL, Kwok JT and Yeung D (2004) Bayesian inference on principal component analysis using reversible jump markov chain monte carlo. In: *Proceedings of the Nineteenth National Conference on Artificial Intelligence, Sixteenth Conference on Innovative Applications of Artificial Intelligence, July 25-29, 2004, San Jose, California, USA*. pp. 372–377.

Appendices

A Index to Multimedia Extensions

Table 2. Index to multimedia extensions.

Ext	Type	Description
1	video	semi-autonomous teleoperation results
2	video	SOSC simulations
3	codes	algorithms, experiments and datasets

B Symbols and Descriptions

Table 3. Description of symbols.

Symbol	Description
ξ_t	observation at time t of dimension D
z_t	hidden state of ξ_t in $\{1 \dots K\}$
a_t	transition matrix with entries $a_{t,i,j}$
$\{\mu_{t,i}^S, \Sigma_{t,i}^S\}$	state duration mean and variance
$\{\mu_{t,i}, \Sigma_{t,i}\}$	output state distribution parameters
$d_{t,i}$	subspace dimension of state/cluster
$\Lambda_{t,i}^{d_{t,i}}$	projection matrix of $d_{t,i}$ eigen vectors
$U_{t,i}^{d_{t,i}}$	$d_{t,i}$ basis vectors of $\Sigma_{t,i}$
b_m	bandwidth parameter to limit cluster size
$w_{t,i}$	weight of parameter set at time t
$g_{t,i}$	projection of ξ_t on $U_{t,i}^{d_{t,i}}$
$p_{t,i}$	retro-projection of $g_{t,i}$ in original space
$R_{t,i}$	rotation matrix to update $U_{t,i}^{d_{t,i}}$
δ_i	distance of ξ_t to each subspace of $U_{t,i}^k$
$\bar{e}_{t,i}$	average distance vector of δ_i
$n_{t,i}$	state transitions count from i till time t
$c_{t,i,j}$	state transitions count from i to j
s_t	duration steps count
λ	penalty for number of states
λ_1	penalty for subspace dimension
λ_2	penalty for transition to less visited states
λ_3	penalty for transition to unvisited state

C Infinite Horizon Linear Quadratic Regulator

The desired reference state $\mathcal{N}(\hat{\mu}_{t_0}, \hat{\Sigma}_{t_0})$ is smoothly followed by using an infinite-horizon linear quadratic regulator with a double integrator system. The cost function

to minimize at current time step t_0 is given by

$$c(\xi_t, u_t) = \sum_{t=t_0}^{\infty} (\xi_t - \hat{\mu}_{t_0})^\top Q_{t_0} (\xi_t - \hat{\mu}_{t_0}) + u_t^\top R u_t,$$

$$\text{s.t. } \dot{\xi}_t = \begin{bmatrix} 0 & I \\ 0 & 0 \end{bmatrix} \xi_t + \begin{bmatrix} 0 \\ I \end{bmatrix} u_t,$$

where $u_t \in \mathbb{R}^m$ is the control input of the system. Setting $Q_{t_0} = \hat{\Sigma}_{t_0}^{-1}$, $R \succ 0$, $\xi_t = [x_t^\top \dot{x}_t^\top]^\top$, $\hat{\mu}_{t_0} = [\hat{\mu}_{t_0}^{x^\top} \hat{\mu}_{t_0}^{\dot{x}^\top}]^\top$ with x, \dot{x} representing the position and velocity of the system, the optimal control input u_t^* obtained by solving the algebraic Riccati equation is given by

$$u_t^* = K_t^P (\hat{\mu}_{t_0}^x - x_t) + K_t^V (\hat{\mu}_{t_0}^{\dot{x}} - \dot{x}_t),$$

where K_t^P and K_t^V are the full stiffness and damping matrices for following the desired reference state.

D Finite Horizon Linear Quadratic Tracking

Consider a double integrator system as an analogue of a unit mass attached to the datapoint ξ_t . The desired step-wise reference trajectory $\mathcal{N}(\hat{\mu}_t, \hat{\Sigma}_t)$ is smoothly tracked by minimizing the cost function

$$c_t(\xi_t, u_t) = \sum_{t=1}^T (\xi_t - \hat{\mu}_t)^\top Q_t (\xi_t - \hat{\mu}_t) + u_t^\top R_t u_t,$$

$$\text{s.t. } \dot{\xi}_t = \begin{bmatrix} 0 & I \\ 0 & 0 \end{bmatrix} \xi_t + \begin{bmatrix} 0 \\ I \end{bmatrix} u_t,$$

starting from the initial state ξ_1 . Let $\xi_t = [x_t^\top \dot{x}_t^\top]^\top$, $\hat{\mu}_t = [\hat{\mu}_t^{x^\top} \hat{\mu}_t^{\dot{x}^\top}]^\top$ where x, \dot{x} represent the position and velocity of the double integrator system. Setting $Q_t = \hat{\Sigma}_t^{-1} \succeq 0$, $R_t \succ 0$, the control input u_t^* that minimizes the cost function is given by

$$\begin{aligned} u_t^* &= -R_t^{-1} B_d^\top P_t (\xi_t - \hat{\mu}_t) + R_t^{-1} B_d^\top d_t, \\ &= K_t^P (\hat{\mu}_t^x - x_t) + K_t^V (\hat{\mu}_t^{\dot{x}} - \dot{x}_t) + R_t^{-1} B_d^\top d_t, \end{aligned}$$

where $[K_t^P, K_t^V] = R_t^{-1} B_d^\top P_t$ are the full stiffness and damping matrices, $R_t^{-1} B_d^\top d_t$ is the feedforward term, and P_t, d_t are the solutions of the following differential equations

$$\begin{aligned} -\dot{P}_t &= A_d^\top P_t + P_t A_d - P_t B_d R_t^{-1} B_d^\top P_t + Q_t, \\ -\dot{d}_t &= A_d^\top d_t - P_t B_d R_t^{-1} B_d^\top d_t + P_t \hat{\mu}_t - P_t A_d \hat{\mu}_t, \end{aligned}$$

with terminal conditions set to $P_T = 0$ and $d_T = 0$. Note that the gains can be precomputed before simulating the system if the reference trajectory and/or the task parameters do not change during the reproduction of the task. The resulting trajectory ξ_t^* smoothly tracks the step-wise reference trajectory $\hat{\mu}_t$ and the gains K_t^P, K_t^V stabilize ξ_t along ξ_t^* in accordance with the precision required during the task.